International Journal of Greenhouse Gas Control 105 (2021) 103223

Contents lists available at ScienceDirect

International Journal of Greenhouse Gas Control

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

Towards a predictor for CO_2 plume migration using deep neural networks

Gege Wen *, Meng Tang, Sally M. Benson

Department of Energy Resource Engineering, Stanford University, 367 Panama St, Stanford, CA 94305-2220, United States

A B S T R A C T

This paper demonstrates a deep neural network approach for predicting carbon dioxide (CO_2) plume migration from an injection well in heterogeneous formations with high computational efficiency. With the data generation and training procedures proposed in this paper, we show that the deep neural network model can generate predictions of CO_2 plume migration that are as accurate as traditional numerical simulation, given input variables of a permeability field, an injection duration, injection rate, and injection location. The neural network model can deal with permeability fields that have high degrees of heterogeneity. Unlike previous studies which did not consider the effect of buoyancy, here we also show that the neural network model can learn the consequences of the interplay of gravity, viscous, and capillary forces, which is critically important for predicting CO_2 plume migration. The neural network model has an excellent ability to generalize within the training data ranges and to a limited extent, the ability to extrapolate beyond the training data ranges. To improve the prediction accuracy when the neural network model needs to extrapolate to situations or parameters not contained in the training set, we propose a transfer learning (fine-tuning) procedure that can quickly teach the trained neural network model new information without going through massive data collection and retraining. With the approaches described in this paper, we have demonstrated many of the building blocks required for developing a general-purpose neural network for predicting CO_2 plume migration away from an injection well.

1. Background

Carbon capture and storage (CCS) is an essential greenhouse gas mitigation technology (IPCC, 2014) which entails capturing carbon dioxide (CO_2) from concentrated sources, the atmosphere, or through bio-energy production, compressing the captured CO_2 into a super-critical fluid, and injecting into deep geological formations for long term sequestration. Once CO_2 is injected into the subsurface formation, the migration of the plume is controlled by the interplay of viscous, gravitational, and capillary forces (Ide et al., 2007; Lee et al., 2016). Since CO_2 has a density that is about 30–40% lower than water, it migrates away from the injection well due to viscous forces while rising upward due to gravitational forces (Pruess and Nordbotten, 2011). Low permeability layers with high capillary entry pressure counteract the effect of gravitational forces by acting as baffles that retain the CO_2 deep in formation (Doughty and Pruess, 2004; Saadatpoor et al., 2010; Krevor et al., 2015). These complex and coupled processes determine the movement of the injected CO_2 in the geological formation, and is critical for project operation design, site characterization, and risk analysis (NETL, 2017). Therefore, the ability to predict the CO_2 plume migration using approaches that take into account these processes is a prerequisite for effective CCS projects.

Currently, numerical simulation is the primary tool used for quantifying the migration processes by solving spatially and temporally discretized mass and energy balance equations for multiphase flow of CO_2 and brine. However, full physics simulations with sufficient grid refinement are often very computationally expensive due to the large spatial domains, long time frames, and multiphysics nature of these problems (NAS, 2018). The inherent uncertainty in the spatial distribution of geological attributes is also a major challenge for numerical simulations, where a single deterministic model is not sufficient to provide accurate predictions of the CO_2 plume migration (Nordbotten et al., 2012). In practice, when dealing with subsurface multi-phase flow problems, methods of history matching (also referred as inverse modeling) (Oliver and Chen, 2011) and uncertainty quantification (Tartakovsky, 2013; Kitanidis, 2015) have been widely studied to increase the confidence of simulation results and aid engineering decisions. The goal of history matching is to use the observed data to estimate and calibrate the geological parameters in an iterative fashion (Strandli et al., 2014; Zhang et al., 2014). It requires a large number of forward simulations, which induces huge computational costs. Uncertainty quantification, on the other hand, evaluates the impact of parameter uncertainty on predicted performance. Conventional uncertainty quantification often relies on Monte Carlo based sampling (Robert and Casella, 2000) to obtain probabilistic estimates, which also requires large numbers of repetitive forward simulations.

Alternative approaches for predicting CO_2 plume migration include reduced physics models which involve only the critical processes and

* Corresponding author.
E-mail address: gegewen@stanford.edu (G. Wen).

https://doi.org/10.1016/j.ijggc.2020.103223
Received 29 May 2020; Received in revised form 19 November 2020; Accepted 29 November 2020
Available online 30 December 2020

some simplifying assumptions (Szulczewski et al., 2012; MacMinn et al., 2011). These models can provide useful preliminary estimates of the CO_2 plume migration and trapping mechanisms, however, cannot be used as a substitute to the full physics numerical simulations that represent the realistic multiphase flow processes in the subsurface formation. In a more general context of subsurface flow investigation, many surrogate modeling techniques have been proposed in literature to improve the computational efficiency (e.g. reduce-order modeling (Cardoso et al., 2008; He et al., 2013), polynomial chaos (Bazargan et al., 2015), Gaussian Process (Tian et al., 2017; Hamdi et al., 2017), etc.). However, most of these surrogate models also requires simplification or reformulation of the problem, while suffering from the curse of dimensionality (Xiu, 2010) and failing to work for high-dimensional inputs such as the heterogeneous formation permeability.

To mitigate these challenges, in this paper, we propose a neural network approach for making fast and accurate predictions of CO_2 plume migration away from the injection well. Neural networks are composed of nonlinear nodes and interconnected layers, which can theoretically estimate any complex function given adequate setup and training (Hagan et al., 1996; Haykin, 1999). In the past decade, the applications of neural networks have exponentially increased due to the development of graphics processing units (GPUs) and data availability (Rusk, 2015). Convolutional neural networks (CNN), specifically, are known for their ability to process image inputs. Convolutional layers with filters can handle localized spatial features while reducing the number of learnable parameters, making neural networks easy to train (Krizhevsky et al., 2012). CNNs are adopted in many science and engineering tasks that involve image data (Vandal et al., 2018; Reichstein et al., 2019; Shen, 2018). Examples of such applications that are closely related to CCS include CO_2 leakage mass detection using seismic data (Zhou et al., 2019), permeability estimation from 2D transmitted light microscopy images (Araya-Polo et al., 2020), and geological probabilistic inversion to generate realizations that fit the data while visually resembling the real world formation (Laloy et al., 2018, 2017; Liu et al., 2018).

In the context of subsurface flow and transport simulation, we have seen some promising applications using CNNs to predict mappings from the input properties (e.g. permeability) to the output states (e.g. flow velocity, saturation, pressure) (Zhu and Zabaras, 2018; Mo et al., 2019a, b, 2020; Zhong et al., 2019). Zhu and Zabaras (2018) first proposed an encoder-decoder network to solve a single-phase flow problem as an image-to-image regression task where the network predicts the flow velocity and pressure fields given a permeability map. Applying a similar encoder-decoder network, Mo et al. (2019b) studied a CO_2–water multiphase flow problem, in which the injection duration is added as an additional scalar input to the network and broadcast to a separate channel in the latent space to make predictions over time. This encoder-decoder network structure was also used in a groundwater contaminant transport inversion problem, in which the time dependence is captured by an autoregressive model (Mo et al., 2019a). A 3D version of this network was published recently where they utilize 3D convolution filters instead of 2D convolution filters to resolve a solute transport problem (Mo et al., 2020). Another approach by Zhong et al. (2019) uses a conditional Generative Adversarial Network to conduct the input–output mapping. Their study also focuses on a CO_2–water multiphase flow problem and the injection duration is included as a conditional number that serves the training as auxiliary information. It is also worth mentioning that some very recent studies attempt to utilize governing equations in the loss function to conduct unsupervised learning, however, these approaches are not yet applicable to multiphase flow problems (Zhu et al., 2019; Geneva and Zabaras, 2020).

Building on previous studies, here we demonstrate some of the essential building blocks for developing a general-purpose neural network algorithm for predicting CO_2 plume migration from an injection well. These building blocks include: accurate predictions of plume migration around an injection well into a storage reservoir with a wide variety of rock properties (permeability and capillary pressure curves) and different degrees of heterogeneity; recognizing different plume migration patterns for a wide range of viscous to gravitational forces; accounting for injection at different depths in the formation; and predicting plume migration for different flow rates and for different durations. Note that the CO_2 plume migration problem is also influenced by other parameters such as porosity, residual gas saturation, and salinity (Song et al., 2020). However, the scope of this paper only focuses on the aforementioned parameters. With the neural network architecture we have developed, we demonstrate that the trained model performs very well at all of these tasks. Our test cases show the model's capability of achieving very high prediction accuracy in a system with complex and discontinuous geological heterogeneity and a wide range of viscous to gravitational forces. The evidence supporting the conclusion is provided by error metrics such as root mean square error (RMSE), mean absolute error (MAE), and mean plume error (MPE).

The body of this paper is organized as follows. Section 2 introduces the problem statement, which includes the governing equations and the numerical simulator used to generate data to train the neural network model. Section 3 describes the methodology for using deep neural networks for multiphase flow prediction, which includes data set preparation, model training, and model prediction. The results are discussed in Section 4, along with a sensitivity study on the size of training data set. In Section 5, we demonstrate the use of transfer learning to improve the model's predictive capability without having to completely retrain the model for problems that are outside of the training data set used to develop the neural network model. In Section 6, we show a comparison of the prediction performance with other neural network algorithms.

2. Problem statement

This paper focuses on a multiphase flow problem with CO_2 in the supercritical phase, and water in the liquid phase. We are interested in a 2D radially symmetrical system that replicates the geometry of CO_2 being injected into a storage reservoir through a vertical injection well. In this section, we first introduce the governing equations for this CO_2–water multiphase flow problem, then discuss the numerical simulator used to generate the data sets for training the neural network model.

2.1. Governing equations

For multiple-phase flow, we have the mass conservation for each component κ described as (Pruess et al., 1999)

$$\frac{\partial M^{\kappa}}{\partial t} = -\nabla \cdot \mathrm{F}^{\kappa} + q^{\kappa}, \tag{1}$$

where component κ is CO_2 or water, M denotes the mass accumulation term, $\mathbf{F}$ is the mass flux, and q is the source term. For each component, the mass accumulation term is summed over phases p,

$$M^{\kappa} = \phi \sum_{p} S_p \rho_p X_p^{\kappa}, \tag{2}$$

where ϕ is the porosity, S_p is the saturation of phase p, ρ_p is the density of phase p, and X_p^{κ} is the mass fraction of component κ presents in phase p.

For each component κ, we also have the advective mass flux $\mathbf{F}^{\kappa}|_{adv}$ obtained by summing over phases p,

$$\mathrm{F}^{\kappa}|_{\mathrm{adv}} = \sum_{p} X_p^{\kappa} \mathrm{F}_p \tag{3}$$

where each individual phase flux $\mathbf{F}_p$ is governed by Darcy's law:

$$\mathrm{F}_p = \rho_p \mathrm{u}_p = -k \frac{k_{\mathrm{rp}} \rho_p}{\mu_p} (\nabla P_p - \rho_p \mathrm{g}). \tag{4}$$

Table 1
ECLIPSE simulation parameters. Note for the capillary pressure curve, $S_{ls} = 0.999$ is used instead of $S_{ls} = 1$ to avoid numerical errors in ECLIPSE simulations.

Parameter	Value/formula
Formation depth	1000 m
Formation thickness	128 m
Porosity	0.2
Temperature (isothermal)	40 °C
Salinity	0
Relative permeability – Corey's curve	
Residual liquid saturation – S_{lr}	0.3
Residual gas saturation – S_{gr}	0
Capillary pressure curve – van Genuchten	
Fitting parameter – λ	0.3
Saturated liquid saturation – S_{ls}	0.999
Maximum capillary pressure – P_{max}	2.1E7 Pa
Capillary entry pressure – Leverett J-function	$P_{\text{entry}} = \frac{\sqrt{k_{\text{ref}}/\phi_{\text{ref}}}}{\sqrt{k/\phi}} P_{\text{ref}}$
k_{ref}	3.95E−14 m^2
ϕ_{ref}	0.185
P_{ref}	7.5E3 Pa

Here $\mathbf{u}_p$ is the Darcy velocity of phase p, k is the absolute permeability, k_{rp} is the relative permeability of phase p, μ_p is the viscosity of phase p, and $\mathbf{g}$ is the gravitational acceleration. The fluid pressure of phase p

$$P_p = P + P_{\text{cp}} \tag{5}$$

is the sum of the reference phase (usually the gas phase) pressure P and the capillary pressure P_{cp}. To simplify the problem setting, our simulation does not explicitly include molecular diffusion and hydrodynamic dispersion.

2.2. Numerical simulation for generating the training data set

The numerical simulation of the above problem is performed by a state-of-the-art full-physics numerical simulator ECLIPSE 300. ECLIPSE 300 uses the finite difference system with upstream weighting and the Adaptive IMplicit method for simulation (Schlumberger, 2014).

This study focuses on CO_2 plume migration away from an injection well using an axisymmetric (rz) model, which is a challenging problem due to the complex interaction between viscous, capillary, and gravitational forces. In particular, as compared to a xy problem, gravity plays an important role and permeability heterogeneity, especially layered heterogeneity, can significantly affect the migration of a CO_2 plume (Wen and Benson, 2019).

The system is radially symmetrical with an injection well located at the center of a cylindrical volume. At the top and bottom boundaries we impose a non-flow condition. The outer boundary in the radial direction is held at a constant pressure. A uniform grid of 128 × 128 elements with a grid size of 1 m is used in the numerical simulation. Each simulation takes ~15 min on an Intel Core i7-4790 CPU and a maximum number of 8 simulations can be run in parallel. Fixed parameters used for the simulations are summarized in Table 1 while variable parameters are discussed in Section 3.1. This system is not intended to realistically represent the geometry of an actual CO_2 storage project, but to allow

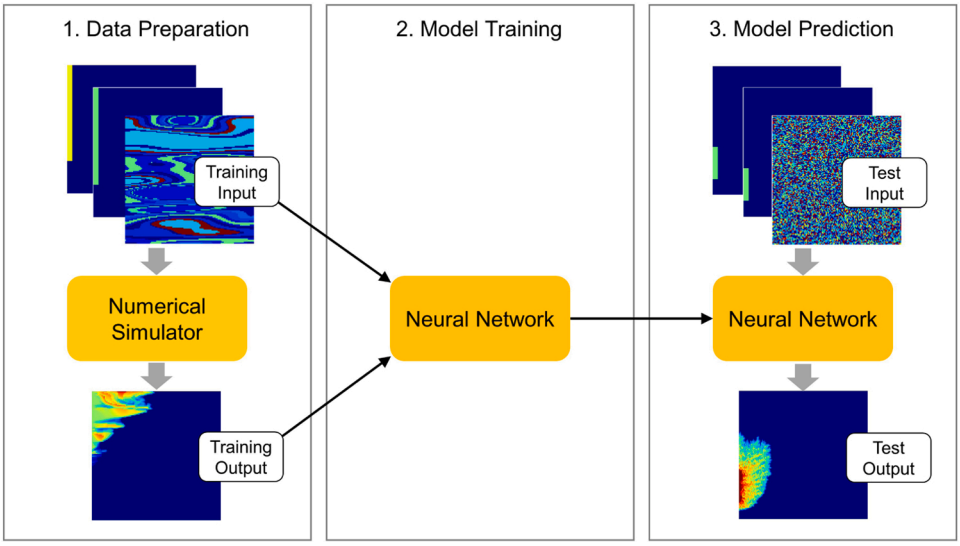

Fig. 1. The workflow of using neural networks to predict multiphase flow results.

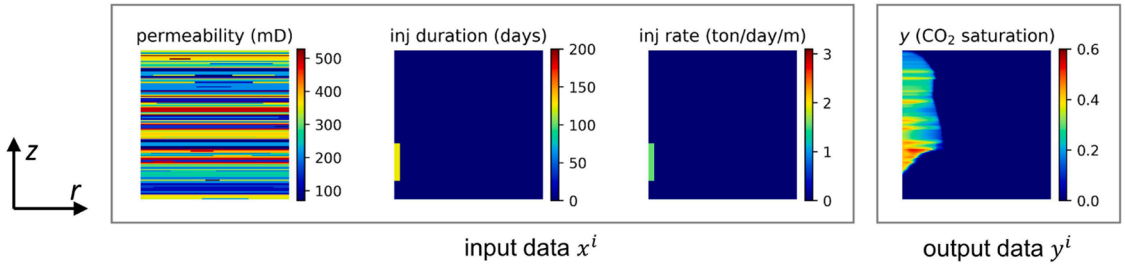

Fig. 2. An example from the training set. The permeability, injection duration, injection rate, and saturation fields all lie in a rz plane, which can be seen as cross sections along the radius. The information about the injection perforation depth is embedded in the injection duration and injection rate maps by the location of the values on the left most column. The modeled volume is radially symmetrical and the injection well is located in the center of the cylindrical volume. The CO_2 saturation is plotted at the end of the injection duration.

Table 2
SGeMS input parameters for random field generation. The lateral heterogeneity correlation length can exceed the field dimension. This is designed in order to generate strong laterally correlated (layered) permeability maps.

Parameter	Unit	Value
Lateral heterogeneity correlation length (ar)	m	Uniform random integer in the interval (1, 256)
Vertical heterogeneity correlation length (az)	m	Uniform random integer in the interval (1, ar)
# of rock materials in each field	–	20
Fraction assigned to each material	–	Uniform random number in the interval (0, 1)
Material mean in each field	mD	Uniform random number in the interval (1, 350)
Material standard deviation in each field	mD	Uniform random number in the interval (1, 200)

efficient development and testing of the essential building blocks of more general models.

3. Methodology

This section introduces some of the necessary procedures for using a deep neural network to perform predictions for CO_2 plume migration, which includes data set preparation; model training; and model prediction. The workflow is illustrated in Fig. 1 and each stage is discussed in a subsection. The data set preparation section introduces the mapping between the training input and the training output, as well as the design of a representative data set. In the model training section, we introduce our neural network architecture; the training loss, as well as the training process. The model prediction section discusses the model's test time specifications and error evaluation metrics.

3.1. Data set preparation

In order to the train the neural network to predict the CO_2 plume migration given the formation geology as well as the injection information, the data set has to contain mappings from the various combinations of a permeability field, an injection perforation depth, an injection duration field, and an injection rate field to the corresponding CO_2 saturation plume field. Fig. 2 shows an example of this mapping.

To mimic the sedimentary formations that are used for CO_2 sequestration, 200 permeability fields are randomly generated using the Stanford Geostatistical Modeling Software (SGeMS) with laterally correlated heterogeneity. SGeMS is an open-source computer package for geostatistical modeling with spatially related variables (Remy et al., 2009). Correlation lengths for the permeability heterogeneity distributions range from a few meters to several hundred meters and input parameters of SGeMs is summarized in Table 2. The formation heterogeneity is radially symmetrical with an injection well located in the center cylindrical volume as shown in Fig. 2. Note that the heterogeneity in the rz plane results in a ringed structure around the well. We do not claim these permeability maps are realistic models of any reservoir, but use them as a way to demonstrate the neural network's predictive ability in highly heterogeneous systems. The reservoir permeability distributions include a wide range of isotropic permeability values between 0.5 mD and 0.8 Darcy. While the density difference between the CO_2 and water remains constant, since the permeability values vary over several order of magnitude, the viscous to gravitational force ratio varies by several orders of magnitude. As an example, for injection into a 500 mD reservoir, the plume migrates quickly to the base of the seal and thereafter migrates in a relatively thin layer near the top of the reservoir. For injection into a 5 mD reservoir, the plume migrates largely horizontally and is retained deep in the reservoir, reflecting the consequences of differing viscous to gravitational forces. The permeability fields studied here are non-Gaussian and

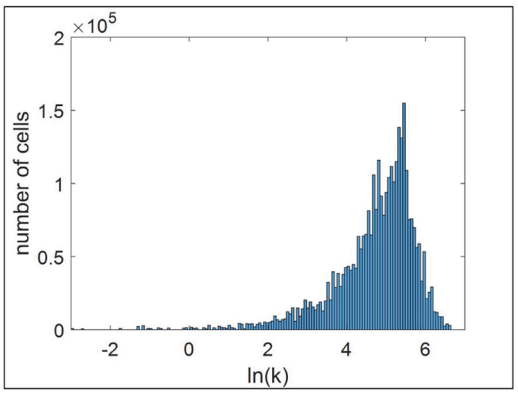

Fig. 3. Permeability histogram of the 200 randomly generated fields being used in the training and test set.

the histogram of permeability values for the grid cells of the 200 fields is shown in Fig. 3.

The data set is composed of numerical simulation runs in which the supercritical CO_2 is injected with various rates, durations, and injection depth intervals. To incorporate the injection information, we create two separate channels which correspond to the injection rate and injection duration respectively. The injection depth interval is treated as a CO_2 source term at the designated grid cell location, which is marked as the colored cells in the injection duration and injection rate examples in Fig. 2. The length of the injection depth interval can be 16, 32, 48, 64, 80, 96, 112, or 128 m. For the examples we have shown here, the location of the injection interval was assigned randomly. In practice, the injection interval will be selected based on the permeability profile at the injection well. Our neural network model has the flexibility to accommodate this. The injection duration can be 25, 50, 75, 100, 125, 150, 175 or 200 days and the injection rate can be 0.8, 1.6, 2.3 or 3.1 ton/day per meters of well perforation. In the example shown in Fig. 2, the injection duration channel shows that the injection lasts for 125 days and the injection rate channel shows that CO_2 is injected at a rate of 1.6 ton/day per meter of well perforation. The injection well is perforated from 16 to 48 m and this spatial information is embedded in both channels. We choose to use the injection rate and duration as input parameters instead of the total volume because the plume behaves differently when the same volume is injected at a different rate. This is illustrated in Appendix A, which also illustrates the ability of the trained neural network model to recognize different plume migration patterns with differing viscous to gravitational forces.

In order to give the neural network a balanced data set, the number of injection duration scenarios, the injection rate scenarios, and the permeability field scenarios need to be comparable. In our data set, we have 200 randomly generated permeability fields, which constrains the number of injection duration fields and injection rate fields to be comparable to 200. As a result, Our data set contains 36 scenarios of different injection location and 8 injection duration, contributing to a total of $36 \times 8 = 288$ scenarios of injection duration fields. Similarly, we have 4 injection rates for each injection location, resulting in a total of $36 \times 4 = 144$ scenarios of injection rate fields. In total, the full data set has $200 \times 36 \times 8 \times 4 = 230{,}400$ samples. At this stage in the process, no effort was made to minimize or optimize the number of training data sets. This issue will be addressed later.

In our data set, each permeability field, injection rate field, and injection duration field are concatenated to become one data input. In convolutional neural networks, each of the fields making up the data input is called a "channel" in the input. This approach is scalable to more controlling parameters with spatial information (e.g. porosity, initial pressure, initial saturation, etc.), which can be easily stacked to the input as additional channels. Note that the design of the input channels

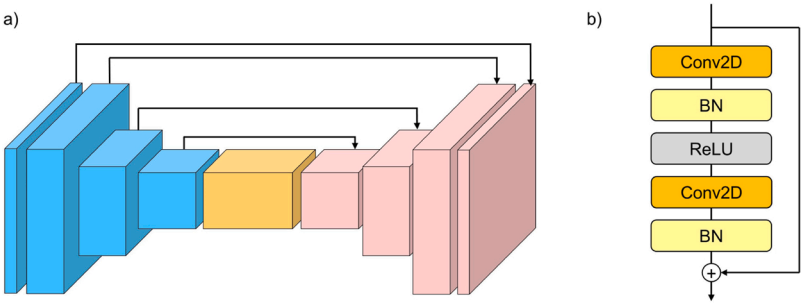

Fig. 4. (a) Schematic of the RU-Net. Blue represents an encoding part, yellow represents a connecting part, pink represents a decoding part, and the black arrow represents a concatenating path. (b) Schematic of a residual convolutional block that makes up the connecting part, which consists a 2D convolutional layer (Conv2D), a batch normalization layer (BN) (Ioffe and Szegedy, 2015), a rectified linear layer (ReLU) (Nair and Hinton, 2010), a second convolutional layer, a second batch normalization layer and an identity shortcut.

is not unique. For example, another possible design for the input is to add a fourth channel which only identifies the injection perforation interval. During training, the learnable parameters in the convolutional filters will be automatically tuned to handle the information in each channel by minimizing the loss function. The benefit of embedding the injection perforation depth into the injection rate and duration channels is that it reduces the number of learnable parameters. Our results shows that the method proposed here is effective even with very small data sets.

3.2. Model training

The following section describes the architecture of the proposed neural network for this CO_2 plume prediction problem. We also discuss the training strategy and the choice of loss function in this section.

3.2.1. Neural network architecture

The main task of the neural network resembles an image to image regression from the input permeability field, injection duration field, and injection rate field to the output CO_2 saturation field. To conduct this task efficiently, we introduce the RU-Net, a deep convolutional neural network architecture developed base on the well-known U-Net architecture (Ronneberger et al., 2015). The U-Net architecture was originally proposed for bio-medical image segmentation, which refers to the process of mapping an input image to an output segmentation mask. Although the application is quite different from the focus of this paper, the U-Net architecture can be effectively adopted here because it is especially efficient in data utilization for the design of the concatenating channel 'highway'. The U-Net architecture contains an encoding path and a decoding path, which captures the hierarchical spatial encoding and decoding of the input and the output images. The concatenating channels pass on the multi-scale spatial information obtained from the encoding path to the corresponding decoding path, where the multi-scale encodings can aid the prediction. Between the encoding and the decoding path, a connecting block is designed to learn the relationship between the input and output latent representations.

Based on the U-Net architecture, in our model, the encoding path and decoding path each contains six encoder/decoder parts as illustrated in Fig. 4a. The latent representations of the input and output are compressed in 128 feature maps with the size of 16x16. Through experiments, we have observed that a larger size and a greater number of the feature maps often requires more trainable parameters while providing higher training accuracy. Therefore, the number and the size of the feature maps are chosen in order to balance the trade-off between training accuracy and training efficiency. To further improve the performance of the deep U-Net architecture in a multiphase flow prediction problem, we used five residual convolutional blocks in the connecting part. The residual convolutional block performs same size convolution with the ResNet framework (He et al., 2016), where a shortcut connection path learns the identity mapping within each part and therefore eases the training difficulties (Fig. 4b).

Notice that the neural network architecture proposed in this paper does not capture the dynamic effect of the time-dependent problem. Instead, we embed the time-dependent variable (injection duration) in the second channel along with the location of the well perforation. The RU-Net architecture can also be combined with the convolutional long short-term memory (LSTM) network in Tang et al. (2020) for capturing dynamic subsurface flow in channelized geological models. However, in this paper, the channel embedding method is used to incorporate the time-dependent variable because it can provide the advantage that the neural network is more compact and easier to train compared to dynamic neural networks. In the meantime, this method does increase the amount of data in the training set. The detailed RU-Net model architecture is described in Table 3.

Table 3
RU-Net architecture. `Conv2D` denotes a 2D convolutional layer; `c` denotes the number of channels in the next layer; `k` denotes the kernel (or filter) size; and `s` denotes the size of the stride; `BN` denotes a batch normalization layer; `ReLu` denotes a rectified linear layer; `Conc` denotes a concatenation; `Unpool` denotes a unpooling layer; and `Padding` denotes a reflection padding layer.

Part	Layer	Output Shape
Input		(128,128,3)
Encode 1	`Conv2D(c16k3s2)/BN/ReLu`	(64,64,16)
Encode 2	`Conv2D(c32k3s1)/BN/ReLu`	(64,64,32)
Encode 3	`Conv2D(c64k3s2)/BN/ReLu`	(32,32,64)
Encode 4	`Conv2D(c64k3s1)/BN/ReLu`	(32,32,64)
Encode 5	`Conv2D(c128k3s2)/BN/ReLu`	(16,16,128)
Encode 6	`Conv2D(c128k3s1)/BN/ReLu`	(16,16,128)
ResConv 1	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(16,16,128)
ResConv 2	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(16,16,128)
ResConv 3	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(16,16,128)
ResConv 4	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(16,16,128)
ResConv 5	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(16,16,128)
ResConv 6	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(16,16,128)
Decode 6	`Conc/Unpool(s2)/Padding/Conv2D (c128k3s1)/BN/Relu`	(16,16,128)
Decode 5	`Conc/Unpool(s1)/Padding/Conv2D (c128k3s1)/BN/Relu`	(32,32,128)
Decode 4	`Conc/Unpool(s2)/Padding/Conv2D (c64k3s1)/BN/Relu`	(32,32,64)
Decode 3	`Conc/Unpool(s1)/Padding/Conv2D (c64k3s1)/BN/Relu`	(64,64,64)
Decode 2	`Conc/Unpool(s2)/Padding/Conv2D (c32k3s1)/BN/Relu`	(64,64,32)
Decode 1	`Conc/Unpool(s1)/Padding/Conv2D (c16k3s1)/BN/Relu`	(128,128,16)
Output	`Conv2D(c1k3s1)`	(128,128,1)

3.2.2. Training procedures

The loss function used for the training of this neural network is the mean square error (MSE), defined as

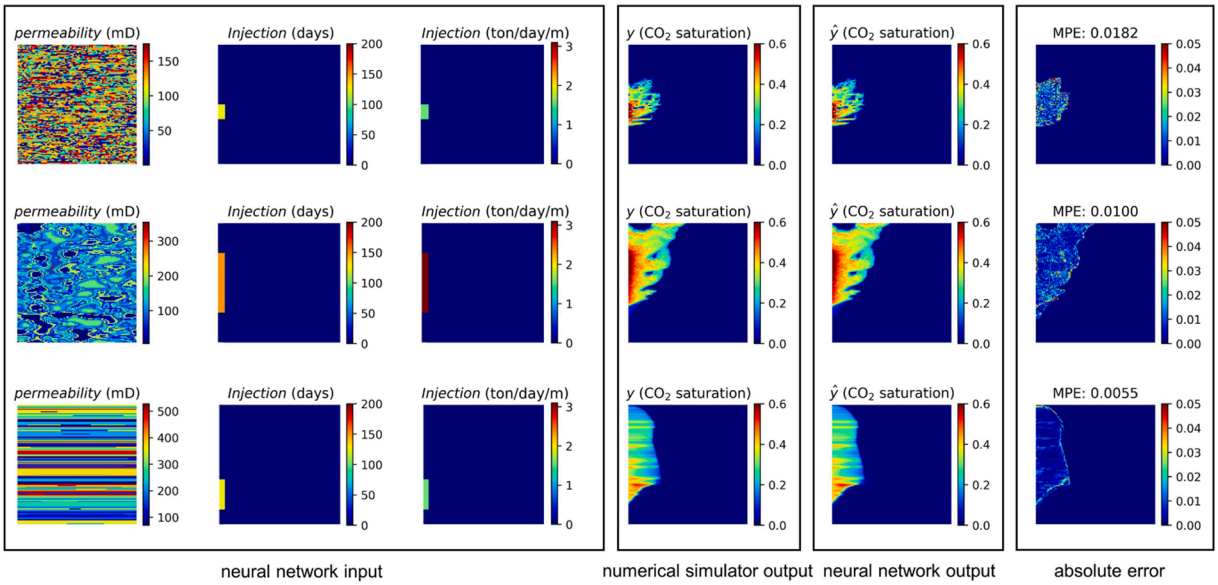

Fig. 5. Test set results on the model that is trained with 200,000 samples for 50 epochs. The 3 examples from the test set demonstrate the model's performance for different combinations of inputs. The first to the third columns shows the input fields for each prediction. The fourth and fifth column show the CO_2 saturation plume simulated by ECLIPSE and predicted by the neural network respectively. The sixth column shows the absolute error on each grid and the MPE is displayed on the title.

$$L_{\mathrm{MSE}} = \frac{1}{N}\sum_{i=1}^{N} ||\mathrm{y}_i - \widehat{\mathrm{y}}_i||_2^2, \tag{6}$$

where N is the number of training samples and $\mathbf{y}$ is the true CO_2 saturation from the numerical simulator. Here $\hat{\mathrm{y}}$ is the CO_2 saturation predicted by the neural network, described as

$$\widehat{\mathrm{y}}_i = \mathrm{f}(\mathrm{x}_i, \boldsymbol{\theta}), \tag{7}$$

where $\boldsymbol{\theta}$ is the learnable parameters in the neural network and $\mathbf{x}_i$ is the input. When given the training data set, $\boldsymbol{\theta}$ is being updated to minimize the loss function and the process conceptually resembles the use of an objective function in inverse problems. The MSE loss especially penalizes errors with larger absolute magnitude compared to those with smaller magnitude. This error is chosen because a higher CO_2 gas saturation in multiphase flow normally corresponds to a higher mobility, which requires a higher accuracy in plume prediction.

In this model, a total of 2,871,841 trainable parameters are updated during the training process to minimize the loss function described above. The update for each learnable weight is based on the gradient to the loss function with respect to the weight (referred as backpropagation in machine learning). A batch size of 16 is selected and the Adam optimizer (Kingma and Ba, 2014) is used with the initial learning rate set to be 0.0001. The details of the batch size and the learning rate hyperparameter selections are included in Appendix B.

To effectively pass the gradient updates to each layer, the neural network needs to be properly initialized with random weights before the training starts. For the convolutional layers, the Glorot normal initializer (also known as the Xavier normal initializers) is used to initialize the kernels (Glorot and Bengio, 2010). In addition, we applied a L2 weight regularizer on each convolutional layer with a hyperparameter of 0.001 to reduce overfitting. Overfitting is a modeling error that occurs when a function is too closely fit to a limited set of data points. Overfitting the model generally takes the form of making an overly complex model to explain idiosyncrasies in the data under study.

During the training process, we use the root mean square error (RMSE) metric to monitor the model's convergence and evaluate the model's prediction performance. The RMSE error is calculated as

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\frac{1}{n}\sum_{i=1}^{N} ||\mathrm{y}_i - \widehat{\mathrm{y}}_i||_2^2}, \tag{8}$$

where N is the number of samples that we are evaluating, n is the number of grid cells of the data output, $\mathbf{y}$ is the true CO_2 saturation from the numerical simulator, and $\hat{\mathrm{y}}$ is the CO_2 saturation predicted by the neural network. The RMSE is a common metric in machine learning to measure model's convergence and a smaller RMSE indicates a more accurate prediction.

3.3. Model prediction

For model prediction, values of the permeability field, the injection rate field, and the injection duration field are fed into the previously trained neural network model. Taking the advantage of the fast prediction ability of a neural network, the model can produce each CO_2 plume output in ~0.0003 s, which is 6 orders of magnitude faster than the state-of-the-art numerical simulator ECLIPSE.

To evaluate the performance of the neural network model, we randomly chose 30,400 data samples from the data set to be the test set, leaving 200,000 data samples for training. The RMSE in the test set is the primary comparison metric to evaluate the performance of a trained model and the test samples are never used in the training of any model. The train/test split ratio is roughly 7/1, which is adequate to provide a representative test result.

In addition, to obtain a more quantitative understanding of the accuracy for each prediction, we use the mean absolute error (MAE) to evaluate the average absolute error on each grid cell. The MAE can be calculated for either a single prediction or for a test set:

$$\mathrm{MAE} = \frac{1}{N}\frac{1}{n}\sum_{i=1}^{N} ||\mathrm{y}_i - \widehat{\mathrm{y}}_i||_1, \tag{9}$$

where N is the number of samples that we are evaluating, n is the number of grid cells of the CO_2 saturation prediction, $\mathbf{y}$ is the true CO_2 saturation output from the numerical simulator, and $\hat{\mathrm{y}}$ is the CO_2 saturation predicted by the neural network.

We also evaluate the mean plume error (MPE), which is especially

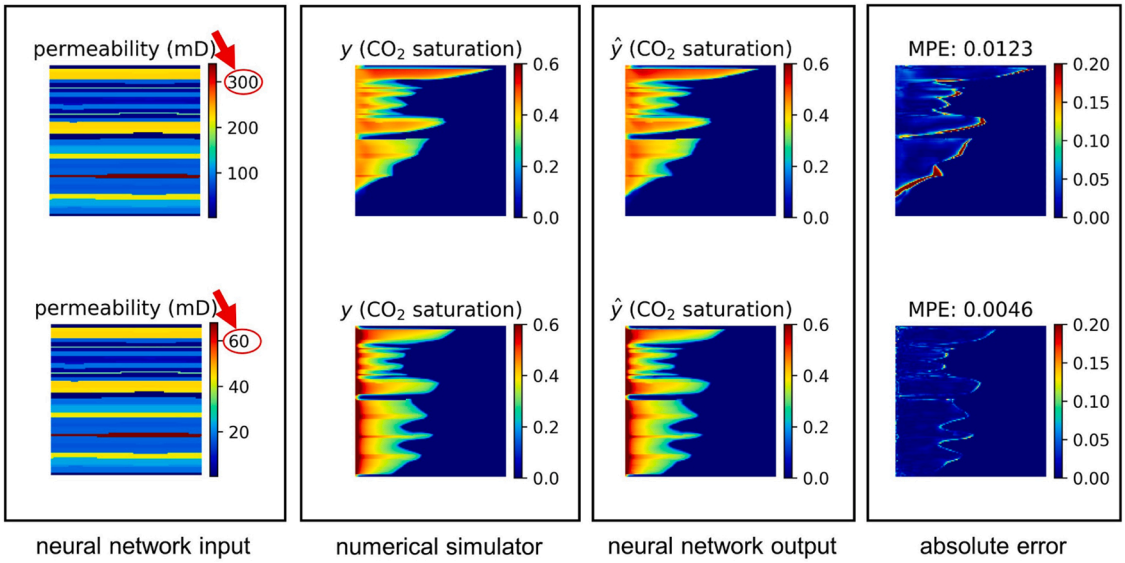

Fig. 6. Comparison between two cases that are identical except that the average permeability in the first case (upper row) is 5 times higher than the second case (lower row). Both cases have fully perforated injection well, with injection duration of 200 days and a rate of 3.1 ton/day/m of perforation. This example demonstrates that the network can deal the complex interplay of viscous, capillary and gravitational forces.

useful for evaluating the prediction accuracy of the CO_2 plume gas saturation. Similarly, the MPE can also be calculated for either a single prediction or for a test set, defined as:

$$\mathrm{MPE} = \frac{1}{N}\frac{1}{n_p}\sum_{i=1}^{N}\|\mathrm{y_i} - \widehat{\mathrm{y}}_\mathrm{i}\|_1. \tag{10}$$

Here n_p is the number of grid cells that has a non-zero value in the true CO_2 saturation output from the numerical simulator $\mathbf{y_i}$. Other symbols represent the same parameters as in MAE.

4. Results

In this section, we first present and analyze the test set CO_2 saturation plume results that are predicted by the neural network model trained with the full data set of 200,000 samples. We then discuss the relationship between the model's performance and the training size through a sensitivity study that compares the test set results from scenarios with different training set sizes.

4.1. Full data set training results

Using the proposed neural network architecture, loss function, and training procedure as discussed in Section 3, we show three examples of the CO_2 saturation predictions in the test set in Fig. 5. Here the neural network model is trained with 200,000 training samples for 50 epochs. In deep learning, an epoch is defined as a single pass of the entire training date set through the neural network. With this large data set, the neural network yields very accurate results for the CO_2 saturation maps. In the test set, the MAE is approximately 0.001, which indicates the average absolute error of CO_2 saturation field prediction on each grid is 0.1%. The average absolute error of the CO_2 saturation within the plume is indicated by MPE, which is 0.009 (0.9%). In the context of multiphase flow simulation and CO_2 plume migration, this error can be considered nearly negligible.

From these examples, we can observe that the trained neural network makes the prediction of CO_2 plume migration according to not only the permeability field, but also the injection duration, injection rate, and well perforation. This verifies the effectiveness of using channels to incorporate variables with spatial information. Moreover, these examples demonstrate that the neural network has an excellent

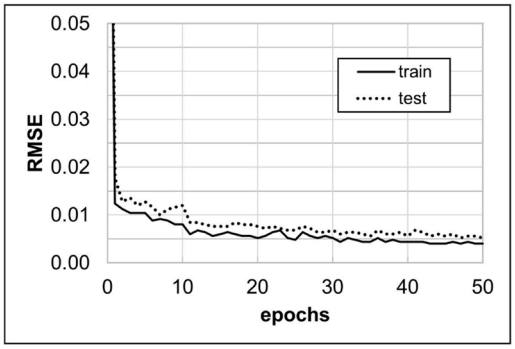

Fig. 7. The evolution of the root mean square error (RMSE) in the training and test set to 50 epochs.

performance with unstructured heterogeneity (first row), channelized heterogeneity (second row), and layered heterogeneity (third row). The errors in the CO_2 saturation prediction mostly occur at the edges of the CO_2 plumes where the true saturation is low. This effect is due to the choice of the loss function, which especially penalizes errors with large values and therefore penalizes to a lesser extent small errors at the edges of the plume.

More importantly, as shown below, we observe that the neural network can accurately capture the consequences of the complex interplay of viscous, capillary, and gravitational forces. In CO_2 storage problems, the migration of a plume is often affected by not only the degree of permeability heterogeneity, but also the absolute value of permeability (Wen and Benson, 2019). In Fig. 6, we show an example where the model predicts plume migration for two different permeability maps that have the same degree of heterogeneity but one of them has an average permeability that is 1/5 of the other. By changing the average permeability in the reservoir, the ratio of viscous forces to capillary and gravitational forces changes, and thus provides a test of whether the neural network can be trained to capture this complexity. Since the capillary entry pressure is scaled using the Leverett J function as described in Table 1, some of the layers become capillary barriers that prevent upward migration of the plume. As shown for the lower permeability case in Fig. 6, due to the effect of these capillary barriers,

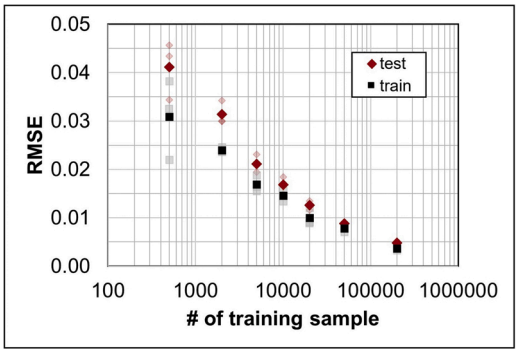

Fig. 8. RMSE on the test set and training set at the end of 50 epochs versus the number of training samples. Each RMSE result is obtained by taking the average of 3 training runs that have different random weight initialization. The test and training RMSE with random weight initializations are represented by light red and light grey dots respectively.

the plume is separated into multiple smaller plumes and more CO_2 is retained at the bottom of the reservoir. Results from Fig. 6 shows that the neural network captures this behavior very well.

The model discussed here was trained for 50 epochs. For the full data set described above, each epoch takes 20–30 min on an NVIDIA Tesla V100 GPU, which leads to a total training time of ~25 h. After 50 epochs, the test set RMSE no longer decreases and the model has converged. In Fig. 7, we show the evolution of the RMSE for both the training and the test set with the number of epochs. The RMSE for both the training and the test set dropped to below 0.2 in just a few epochs. Comparing the final RMSE of the test set to the training set, we observe that the neural network model slightly overfits to the training set. In neural network training, a small degree of overfitting shows that the model performs slightly better on seen data compared to unseen data, which indicates a healthy training procedure.

4.2. Sensitivity to the number of training samples

In this section, we study the influence of the training data set size on the model's performance by conducting a sensitivity study on the number of training samples, where the size of the training set varies from 200,000 to 500 samples. The relationship between the number of the training samples and the test and training set RMSE at the end of 50 epochs is plotted in Fig. 8. Since the training result of a neural network can be sensitive to the random weight initializations, especially when the training set size is small, each of the RMSE results showing in Fig. 8 is obtained by taking the average of 3 training runs that have different random weight initializations.

In general, increasing the number of the training samples monotonically reduces the RMSE. Notice that the relationships between the training set RMSE and the number of training samples show a log-linear behavior, which indicates that reducing the RMSE becomes more and more difficult when the error is small as it requires an exponentially larger number of samples in the training set. We also observe that the difference between the test set RMSE and the training set RMSE increases as the number of training samples decreases, which indicates that training with smaller data sets causes more overfitting.

To further investigate the neural network's performance with different sizes of the training set, three randomly picked test set predictions are shown in the first three rows of Fig. 9. Predictions are compared from 200,000 down to 500 training samples. From the three examples shown in Fig. 9, we can notice that the plume always migrates upward towards the top of the formation, indicating that the neural network can easily learn the effect of gravity even with the smallest training set. Also, the CO_2 is always predicted to migrate away from the location of the well perforations given in the input of the injection field. These results indicate that the technique for embedding the injection perforation depth in the injection rate and injection duration channels is effective even with a very small number of the training samples.

In the fourth to sixth rows of Fig. 9, we compare the absolute errors for each of these cases. We find that that most of the improvement from using a larger training set occurs at the plume edges where the CO_2 saturation is low. This effect is caused by the choice of loss function which especially penalizes larger errors. A previous study has shown that a so-called segmentation loss can be used together with the MSE loss to improve the prediction accuracy on the edges of the CO_2 plume ((Mo et al., 2019a)). Therefore, when training with small data sets, this method can be considered to improve the model's performance.

For each of the training sizes shown in Fig. 9, we also compare the probability density function of the neural network predicted CO_2 plume migration result with the probability density function of the data (Fig. 10). From these comparisons, we can observe that the output predicted by the 200,000 training sample model has a highly accurate probability density function that matches the data. However, the neural networks trained with smaller training set sizes tend to produce saturation outputs with a probability density function distribution that is smoother than the data.

5. Generalization and transfer learning

In the previous section, we demonstrate that the trained neural network can make accurate predictions of the CO_2 saturation maps given the input of the permeability and injection field. Sensitivity studies on the number of training samples show that the neural network can efficiently utilize the training set and achieve low training and test error even with small data sets. However, a key challenge in predicting CO_2 plume with neural networks is whether the neural network can generalize outside of its training data. In this section, we address this concern with the following generalization studies and a demonstration of a transfer learning procedure.

5.1. Generalization

To investigate the ability of the neural network to make predictions for parameters not included in the training set, we studied 4 cases. The cases include generalizing for injection durations intermediate between the durations included in the training set; injection durations longer than the largest used in the training set; new injection depth intervals (interval of the well perforation) that are not included in the training set; and new permeability maps that are not included in the training set. For the purpose of demonstration, we use a fixed injection rate of 2.3 ton/day/m of perforation in all of the following generalization studies and reduce the number of channels in the input to 2 channels with a permeability field and an injection duration field. The size of the training set is 20,000 and the test set were 200 for each of the 4 generalization cases. To provide a performance reference to these generalization cases, we also run a neural network model on 20,000 training samples with a fixed injection rate of 2.3 ton/day/m of perforation for 50 epochs. The training and the test sets MPEs are 1.4% and 1.7%, and the RMSE are 0.007 and 0.010 respectively.

This work demonstrates the ability to generalize for the 4 parameters selected (permeability distributions, injection rate, injection duration, and injection depth interval), to demonstrate the proof of principle that our trained neural network could develop the functionality needed to capture the key processes affecting CO_2 plume migration. Note that in

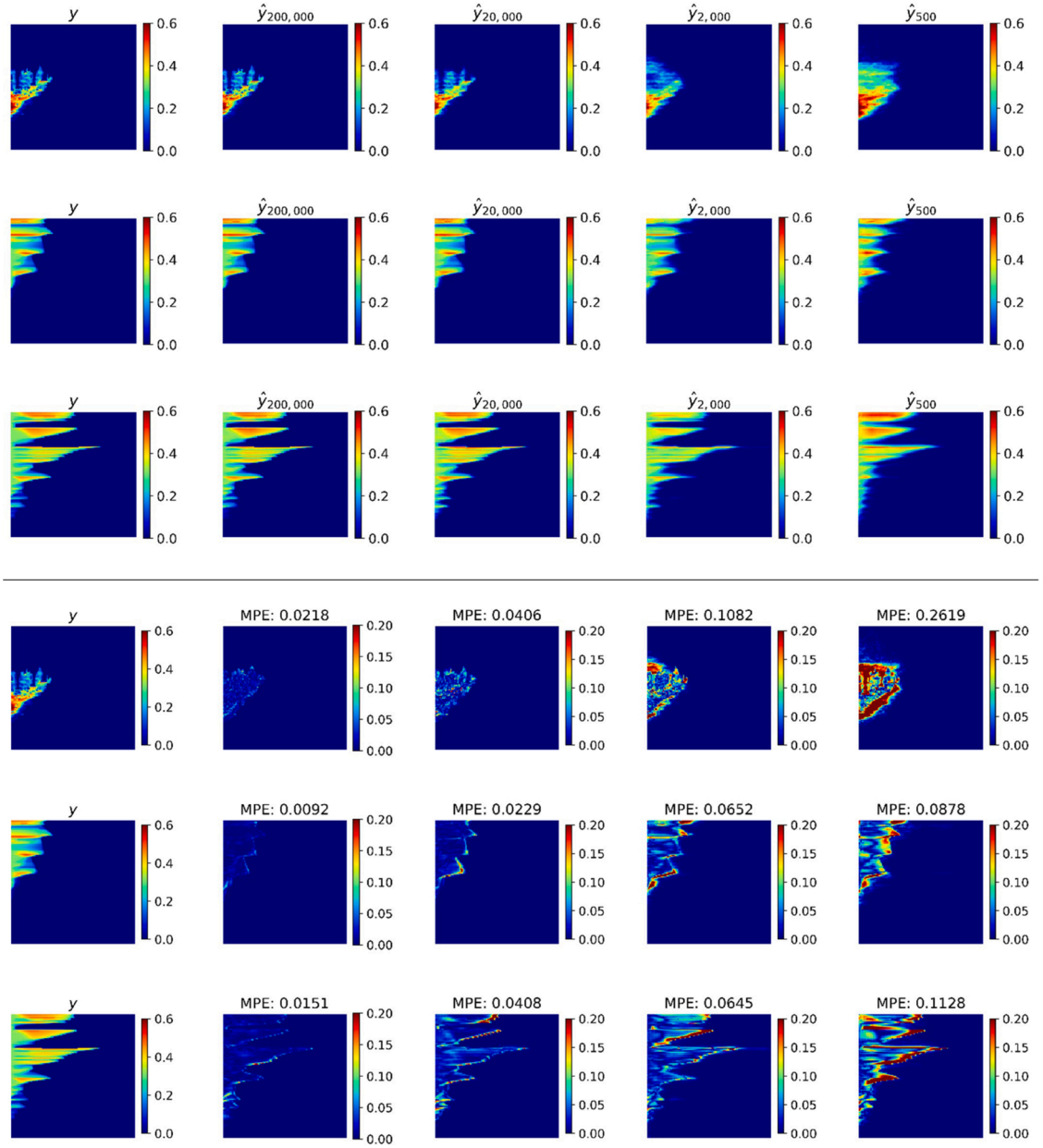

Fig. 9. Row 1–3: CO_2 saturation plume predictions in the test set from the model trained with different sizes of the training samples. Row 4–6: Absolute error visualizations on the test set from the model trained with different sizes of the of training samples. The black bar on the left most column is a schematic of the well and the short horizontal lines marks the location of the perforation interval.

this work we use the neural network to generalize for a parameter only when the parameter is already represented in the data input in some form. For example, for the case where the injection duration in the test is longer than the largest duration used in the training set, the parameter injection duration itself is already embedded in the injection duration channel. For parameters that are not represented in the data input (e.g. salinity, formation temperature, porosity, etc.), additional training would be required.

5.1.1. *Generalization for intermediate injection durations*

To demonstrate the interpolation ability of the neural network to interpolate for injection durations that are intermediate between the durations included in the training set, we use a case where the training set includes samples with 25, 75, 125, and 175 days of injection, and a test set that includes samples with 50, 100, and 150 days of injection. Fig. 11a shows the evolution of the RMSE in the training and test set with an increasing number of epochs for this case and the reference case. The training set RMSE in both cases are similar, reaches to ~0.007 after 50 epochs.

For the test set, the RMSE for this interpolation case is about 0.014 at the end of 50 epochs. An example from the test set is shown in the first row of Fig. 12a, in which we can observe a relatively accurate prediction with some errors on the edges of the plume. The average test set MPE for this case is slightly higher than for the reference case, which is 1.9%. Overall, the neural network has an excellent ability to generalize for intermediate injection durations that are not included in the training set.

5.1.2. *Generalization for longer injection durations*

Similarly, the ability to make predictions for injection durations longer than the largest in the training set is demonstrated by a case where the training set includes data with 25, 50, 75, 100, and 125 days of injection, and a test set includes data with 150, 175, and 200 days of

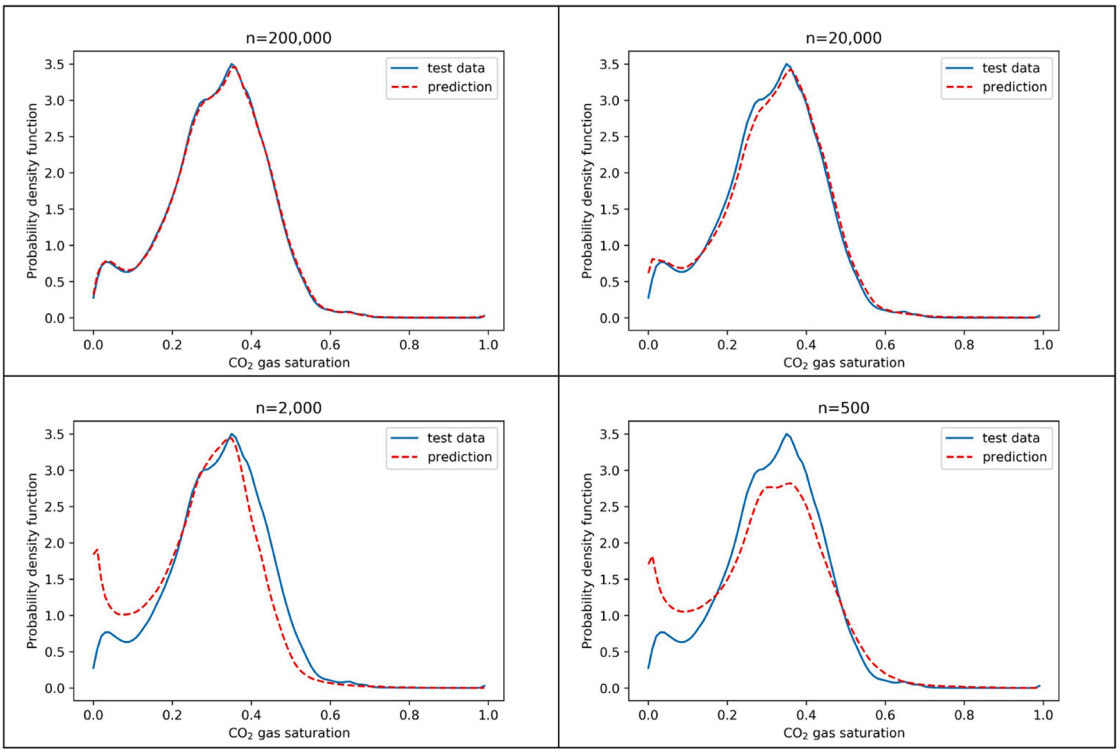

Fig. 10. CO_2 saturation probability density function of the test set data in comparison with the output predicted by the model trained with different sizes of the training samples.

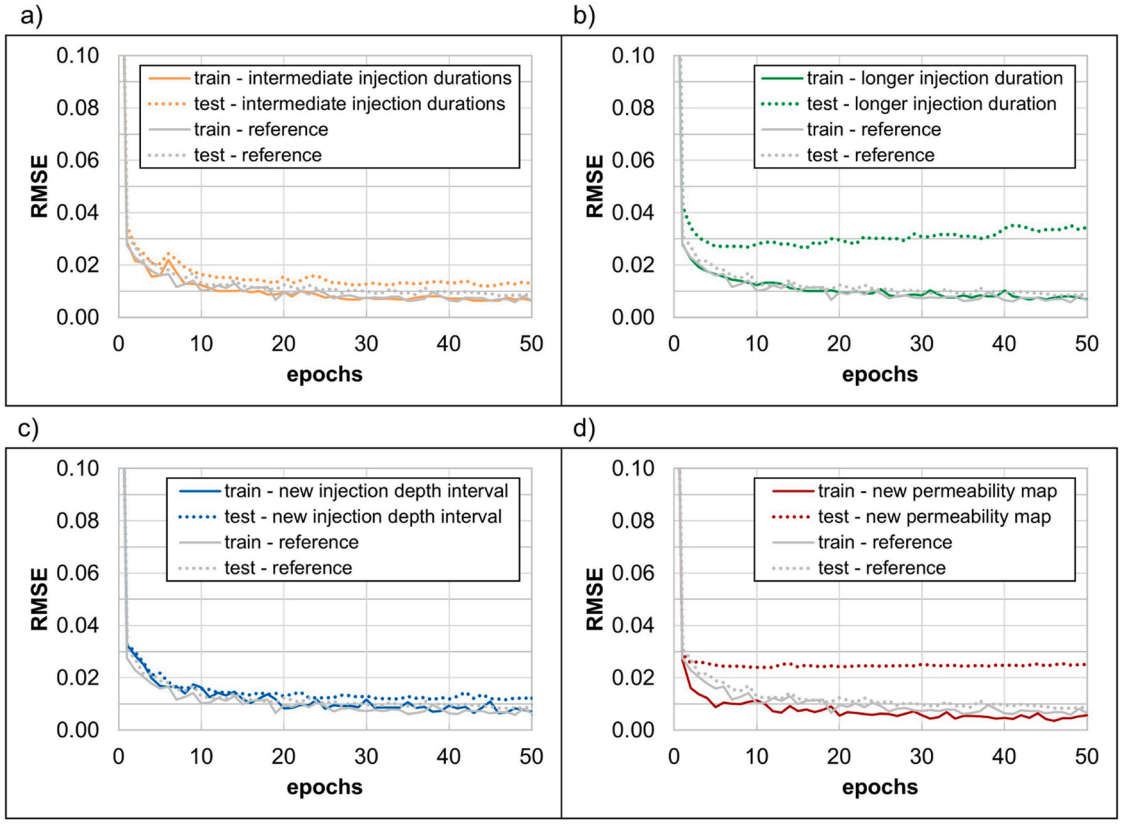

Fig. 11. The test and training set RMSE evolution with the number of epochs. Subplot correspond to the cases generalizing for (a) injection durations intermediate between the durations included in the training set, (b) injection durations longer than the largest used in the training set, (c) different injection depth intervals (interval of the well perforation), and (d) permeability maps not included in the training set.

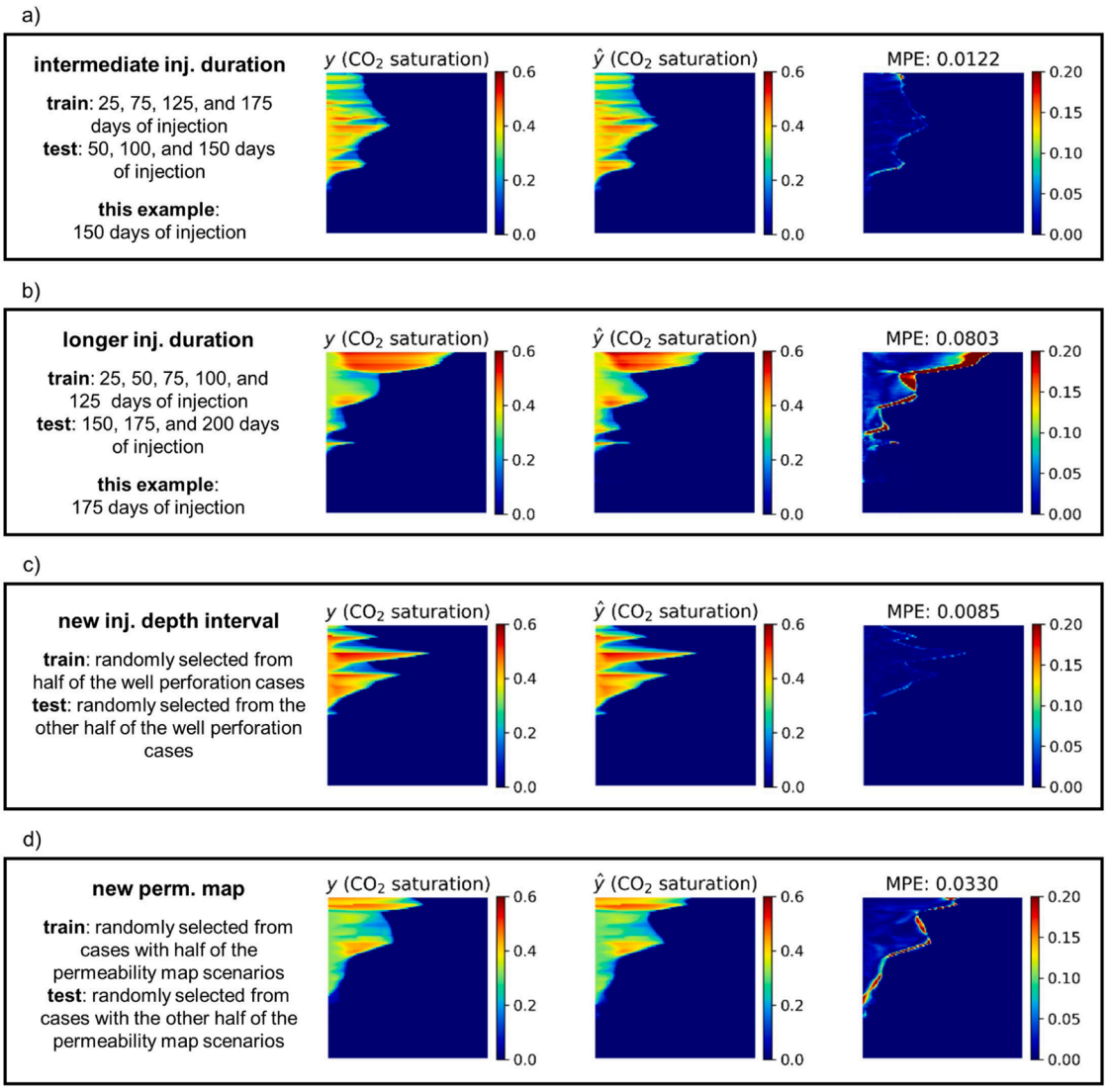

Fig. 12. Subplot (a) shows an example of the test set result where the network interpolates for intermediate injection duration between the injection durations included in the training set. Subplot (b) shows an example of the test set result where the network extrapolates on the injection durations that are longer than the training injections in the training set. Subplot (c) shows an example of the test set result where the network generalize for different injection locations. Subplot (d) shows an example of the test set result where the network needs to generalize for new permeability maps. The first column shows the numerical simulation outputs, the second column shows the neural network predicted outputs, and the third column shows the absolute errors.

injection. In other words, this case requires the neural network to extrapolate on the injection durations. In Fig. 11b, we can observe that the evolution of the training set RMSE is very similar to the reference case.

However, for the test set, we can observe a significantly different behavior of the RMSE, which reaches to ~0.034. Not only is test set RMSE significantly higher than the training set at the end of 50 epochs (~4 times higher), it also shows an increasing trend as the number of epochs increase. This indicates a significant overfitting to the training data, where the network is more and more likely to predict a smaller CO_2 plume that is similar to the training data. The second row of Fig. 12b shows an example from the test set where the neural network predicts a significantly smaller plume and yields an MPE of 8%.

As we will show in the next section, the prediction accuracy can be improved when extrapolation is needed by transfer learning with a limited number of training data. We can also utilize "early stopping" to prevent overfitting by stopping the training procedure when the test set RMSE stops decreasing. In this case, we can stop the training at the 5th epoch instead of the 50th. By using the early stopping technique, the test set RMSE drops from 0.034 to 0.027 and the test set average MPE drops from 5.1% to 4.8%.

5.1.3. *Generalization for the injection depth intervals*

In this case, we investigate the ability for neural network to generalize for injection depth intervals that are different from the injection depth intervals included in the training set. This is demonstrated by using a training set that contains data with half of the randomly selected well perforation intervals, and a test set that contains data with the other half of the well perforation intervals. In both the training and the test set, the injection well has a single and continuous perforation interval and the injection rate is fixed at 2.3 ton/day/m of perforation.

As shown in Fig. 11c, the RMSE for the training and the test sets after 50 epochs are 0.007 and 0.013 respectively. The difference on the RMSE between the training and test set is the smallest among all the generalization cases. The average test set MPE is 1.9%, which is close to the average test set MPE of the reference case (1.7%). This indicates that the model has excellent generalization ability on spatial information like where the CO_2 is injected due to the design of the RU-Net architecture,

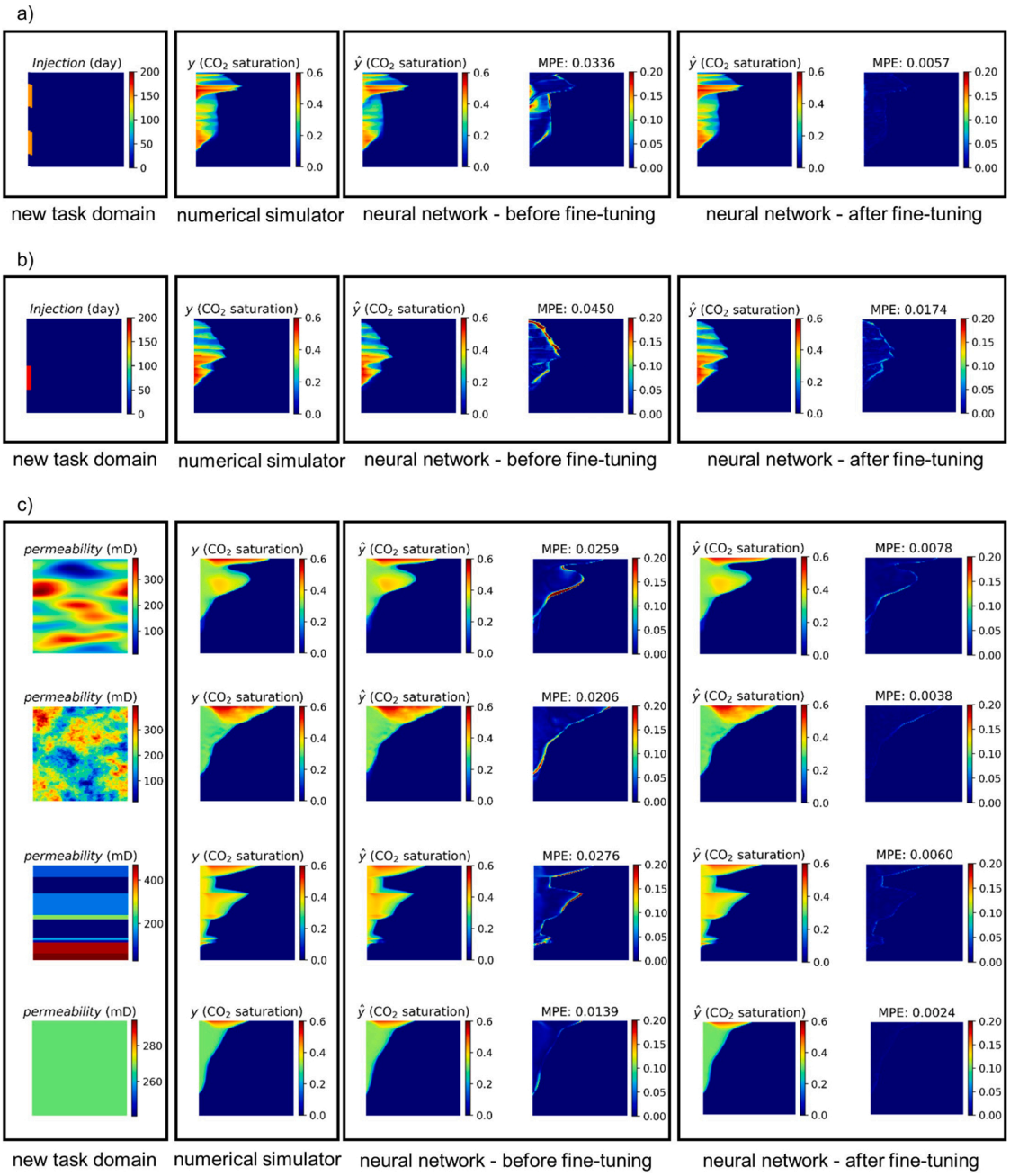

Fig. 13. Examples of neural network fine-tuning with data that has multiple injection well perforations (subplot a), longer injection durations (subplot b), and new permeability map distributions (subplot c). For subplot (a),(b), and (c), the first column shows the inputs from the new task domain. The second column is the true saturation field simulated by the numerical simulator. The third and fourth column are the prediction by the network and the absolute error before applying fine-tuning. The fifth and the sixth column are the prediction by the network and the absolute error after applying fine-tuning.

which can effectively utilize the hierarchical spatial information of the training data. An example from the test set is shown in Fig. 12c.

5.1.4. Generalization for the permeability maps

Another case that we investigated is the model's generalization ability for new permeability maps that are different from the training data. This ability is especially important when using the neural network models as surrogate models in the context of uncertainty quantification, because the neural network is often required to provide predictions on a large number of new permeability maps. To investigate this ability, we train the neural network using only half the permeability maps, randomly selected, and test it using data sample from the other half of the permeability maps.

For this generalization case, the average test set MPE is 3.6%, which is ~1.1 times higher than the average test set MPE for the reference case. Fig. 12d shows a representative example from the test set where the MPE is 3.3%. In addition, Fig. 11d shows that the RMSE for the training and the test sets are 0.006 and 0.024 at the end of the 50 epochs. The test set RMSE is 3 times higher than for the training set, which suggest over-fitting to the training set. Similar to the case where the neural network needs to extrapolate on injection duration, we can use transfer learning techniques to improve the predictions when generalizing for new permeability maps.

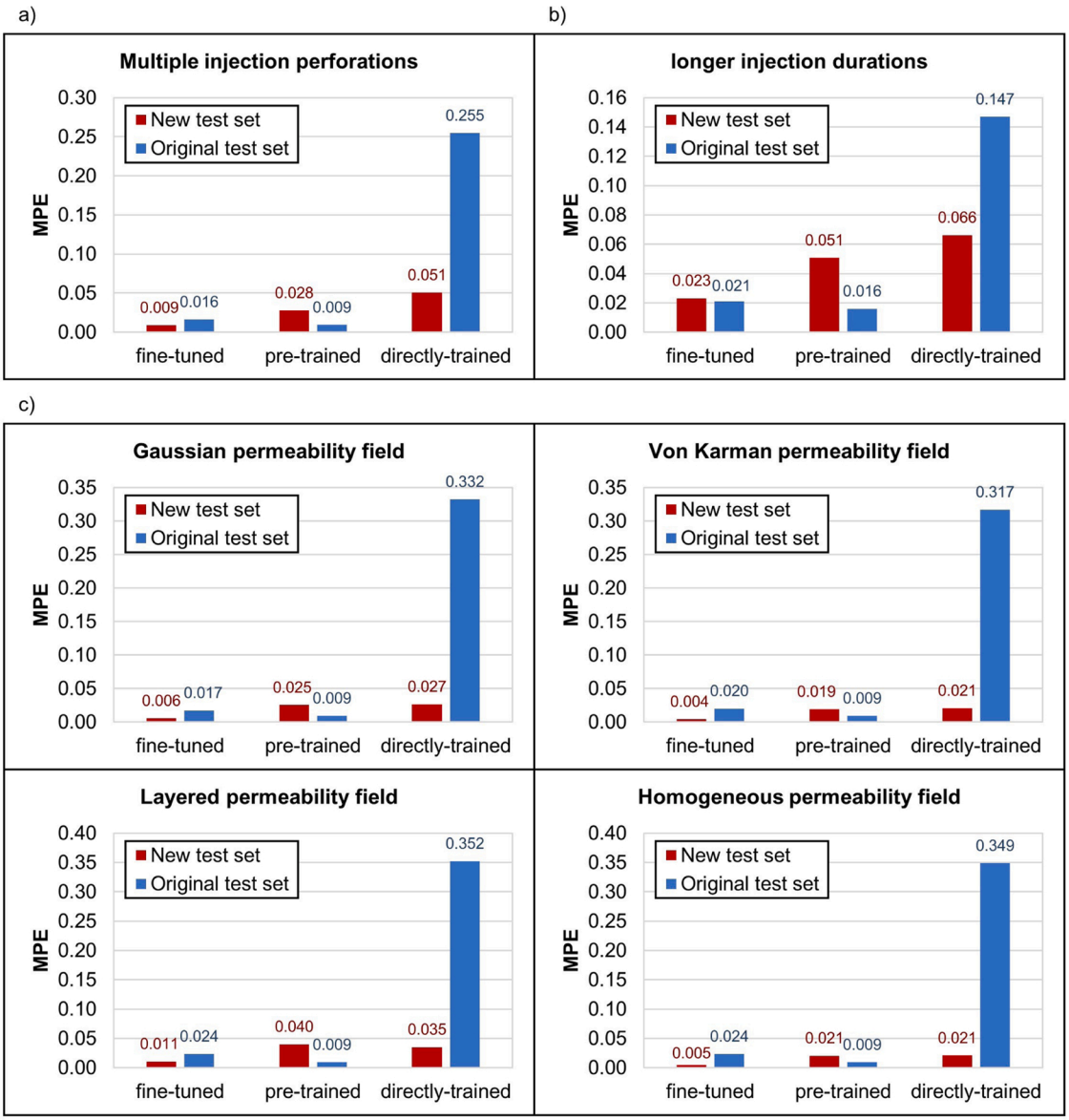

Fig. 14. Each plot includes MPE comparisons by using the fine-tuned model, the pre-trained model, and the model that is directly-trained with the new fine-tuning set. The red bar represents the MPE on the new test set and the blue bar represents the MPE on the original test set. Subplot (a), (b), and (c) corresponds to the new task domain discussed in the Scenario 1, 2, and 3 in Section 5.2.

5.2. *Transfer learning to site specific tasks*

In machine learning, when the training and test data come from different task domains, transfer learning can serve as a powerful tool for improving performance without having to go through the original massive data collection process again (Qiang Yang and Pan, 2010; Radenovic et al., 2018). In the context of CO_2 plume migration, although the data produced by multiphase flow numeral simulators always satisfy the mass and energy balances equations, discrepancy between the training and test data can occur due to specific site properties such as injection pattern, formation depth, formation temperature, or permeability field distribution, etc. Therefore, to transfer the knowledge that a neural network has previously learned from a large data set, we can apply a fine tuning procedure that uses the pre-trained model as the parameter initialization and then conducts training with only a few data samples from the new task domain. Using this technique, we can broaden the applicability of the neural network model and improve the prediction accuracy when the neural network needs to generalize outside of the training data domain. In the following section, we introduce 3 scenarios where the transfer learning technique is applied. For the purpose of demonstration, the injection rate for the following cases are all fixed at 2.3 ton/day/m of perforation.

5.2.1. *Scenario 1 – multiple injection perforation intervals*

In this section, we demonstrate the feasibility of the transfer learning technique by using an example where the neural network needs to predict CO_2 plume migration results when the injection well has multiple perforation intervals. The network that we previously discussed is trained with samples where all of the injection wells have one single and continuous perforation. In reality, injection wells may have multiple sets of well perforations to satisfy operational needs. With multiple well perforation intervals, the injected CO_2 will form multiple plumes that may or may not merge with each other. If the lower CO_2 plume gets connected to the upper plume, gravity will cause the CO_2 to migrate upward and increase the CO_2 saturation of the upper plume. The first column of Fig. 13a shows an example of the injection wells with multiple

perforation intervals, and the second column of Fig. 13a shows the numerical simulation result corresponding to this perforation scenario.

Before fine tuning, we run the trained neural network model with inputs containing multiple perforated intervals. Here we use the neural network that was trained with 200,000 training samples with a single perforation interval. This gives us the prediction and errors shown in the third and fourth columns of Fig. 13a. We can observe that this neural network has a poor prediction accuracy on CO_2 saturation when the CO_2 plumes injected from the two perforation intervals merge together. The MPE on the example presented here increased to ~3.4%, which is more than 2 times larger than the average MPE that we have previously achieved on the test set with a single well perforation interval.

Then, to 'teach' the neural network model how the injected CO_2 plumes behave with multiple perforation intervals, we generated a small training set that includes 80 training samples with multiple perforation intervals. Using the pre-trained model as initialization, the neural network was trained (fine-tuned) on this small training set for only 20 epochs. The fifth and sixth column of Fig. 13a show that the model performs very well after fine-tuning and the MPE drops to 0.6% in this example.

Without using the transfer learning technique, one would have to either directly train the neural network with the small data set that is available, or go through a computationally expensive data collection process. In Fig. 14a, we compared the MPE in the multiple injection perforation test set using the fine-tuned model, the pre-trained model, and the model that is directly-trained with 80 samples in the new task domain. From this comparison, we can observe that the fine-tuned model can achieve the lowest test set average MPE of 0.9% while the directly-trained model yields the highest test set average MPE of 5.1%.

Another advantage of using the transfer learning technique is that the model after fine-tuning still retains the ability to make predictions in the original task domain. From Fig. 14a, we can observe that the fine-tuned model maintains a relatively low MPE (1.6%) on the original test set. When applying the transfer learning technique, the number of epochs in the process of fine-tuning is a hyperparameter that we can tune to get the desirable accuracy in both the original and the new task domains. More epochs of fine-tuning will leads to lower MPEs in the new test set and higher MPEs in the original test set. Finally, we can observe from Fig. 14a that the model directly trained by the 80 samples has very high MPE and cannot be used to make prediction on the original test set.

5.2.2. Scenario 2 – injection duration extrapolation

In this example, we apply the transfer learning technique to a scenario where the neural network needs to extrapolate injection duration beyond the initial training data set (described in Section 5.1.2). The pre-trained model is trained using data samples with injection durations ranging from 25 to 125 days. However, the new task domain contains data samples with injection durations of 125–200 days. Applying the pre-trained model on the new task domain generally leads to a smaller plume and yields an average test set MPE of 5.1%. The third and fourth columns of Fig. 13b shows an example of this behavior.

To apply the transfer learning technique, we generated a new data set that contains 450 samples with the injection duration ranging from 125 to 200 days. Using the new data set and the pre-trained model as initialization, the neural network is fine-tuned for 40 epochs. After fine-tuning, the MPE for the example in Fig. 13b dropped more than 60% compared to the MPE before fine-tuning.

In Fig. 14b, we show the comparison of the MPE on the new test set and the original test set by using the fine-tuned, pre-trained and directly-train models. Again, the fine-tuned model yields low MPEs on both the new and the original test set, while the model that is directly trained by 450 samples from the new task domain performed poorly on both test sets.

5.2.3. Scenario 3 – new permeability map distributions

In this section, we use transfer learning to improve prediction accuracy on 4 types of permeability maps that are not included in the training set. The first column of Fig. 13c shows examples of the 4 types of new permeability maps: a Gaussian correlation field, a von Karman correlation field, a purely layered field, and a homogeneous field. Using the pre-trained model to predict CO_2 saturation outputs on these permeability maps, the model yields MPEs that are 2 to 4.5 times as the MPE in the original test set (Fig. 14c).

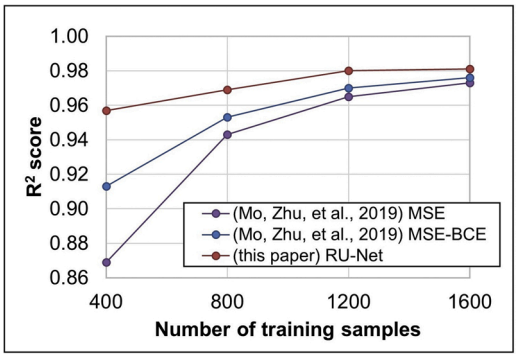

Fig. 15. R^2 comparison (Mo et al., 2019b) with different training sizes. All results were trained for 200 epochs. MSE denotes a neural network that is trained by the mean square error; MSE-BCE denotes a neural network that is trained by a combination of mean square error and binary cross entropy error.

For each type of permeability map, we generated a fine-tuning set that contains 200 data samples. In the fine-tuning set, the injection durations of each of the data samples are fixed at 200 days with fully perforated intervals so that each data sample corresponds to a unique permeability map. We then fine-tuned the neural network for 40–100 epochs depending on the RMSE evolution to achieve low MPE in the new test set while maintaining a good performance in the original test set. As shown in the examples in Fig. 13c and the MPE comparisons in Fig. 14c, after fine-tuning, the test set MPEs on the new permeability maps reduce significantly. At the same time, the fine-tuned models still yield relatively low MPEs on the original test set.

In conclusion, the 3 scenarios discussed in this section demonstrate the potential of using the fine-tuning technique to broaden the applicability of a pre-trained neural network model without going through massive data collection. The experiments also suggest that the fine-tuning technique works effectively and efficiently on task domains that are mainly involved with new spatial information, such as new permeability map distributions and injection well perforation patterns. For task domain that requires temporal extrapolation (injection duration), the improvement of applying the transfer learning technique is less significant and the fine-tuning process requires more data. Nevertheless, the amount of data required in the transfer learning procedure is still significantly less than collecting the full data set.

6. Comparison with other neural network algorithms

Since the main focus of this paper is on the fast and accurate predictions of CO_2 plume migration, the neural network described above does not consider the prediction of pressure outputs in multiphase flow simulations. Nonetheless, the RU-Net proposed in this paper can be easily modified in order to enable pressure prediction. In this section, we demonstrate this idea through a comparison of our RU-Net with the DenseED network used in Mo et al. (2019b). The comparison is performed based on the 50×50 permeability map and CO_2 gas saturation data set that Mo et al. (2019b) provides in their paper.

To train the RU-Net using the data set provided by Mo et al. (2019b), we first padded zeros to increase the dimension of the data to 64×64, so that the data can go through the downsize 3 times during encoding without losing too much information. In this data set, the duration of

injection is incorporated in the second input channel during the training of the RU-Net. Since this data set has a constant injection rate, the channel which represents injection rate field is simply eliminated from the data input and the network can automatically adjust itself to incorporate the differences in the depth of the input. This data set has 2 outputs which are CO_2 gas saturation field and pressure field. To adjust the RU-Net to predict both fields, we changed the last layer of the neural network structure to outputs 2 channels. In addition, we increase the depth of the first encoding layer and the last decoding layers as well as the number of ResConv blocks to deliver the extra information required by the pressure field prediction. Other training hyperparameters are unchanged. The details of the modified neural network architecture are included in Appendix C.

The coefficient of determination (R^2) score is used for this comparison, defined as

$$R^2 = 1 - \frac{\sum_{i=1}^{N} ||\mathrm{y}^i - \widehat{\mathrm{y}}^i||_2^2}{\sum_{i=1}^{N} ||\mathrm{y}^i - \overline{\mathrm{y}}^i||_2^2}, \tag{11}$$

where N is the number of samples in the test set; $\mathbf{y}^i$ is the true CO_2 saturation field; $\widehat{\mathrm{y}}^i$ is the CO_2 saturation predicted by neural networks; and $\overline{\mathrm{y}}^i$ is the average of the true CO_2 saturation field in the test set. The R^2 score is a normalized metric and can be used to compare the prediction accuracy across different data sets. Fig. 15 summarizes the R^2 score with different training sizes.

This comparison shows that our RU-Net achieves higher R^2 at all training sizes. The number of epochs in this case (200) is larger than in the previous cases because the number of data sample is smaller than our data set, which leads to fewer gradient updates each epoch. We observe that the RU-Net is especially efficient at utilizing the data set and achieves good performance when the data set is small. This is due to the design of the concatenating channel 'highway' which pass on the multi-scale spatial information obtained from the encoding path to the corresponding decoding path to aid the prediction.

In addition, notice that Mo, Zhu, et al.'s data set lies on the *xy* plane (no gravity) while our data set lies on the *rz* plane (with gravity). Despite the differences in the number of outputs and physical environment, we can still robustly apply the RU-Net to Mo, Zhu, et al.'s data set without complicated hyperparameter tuning. This comparison validates the performance of the RU-Net that we propose, as well as the robustness of using this neural network on different multiphase flow settings.

7. Conclusion

In this paper, we demonstrate some of the essential building blocks for developing a neural network model for making predictions of CO_2 plume migration in heterogeneous reservoirs from an injection well. The neural network is trained on simulation outputs from a full physics multi-phase flow simulator that includes the effects of gravitational, capillary, and viscous forces in 2D radially symmetrical settings. We find that using the neural network architecture and training procedures described here, the network can provide highly accurate predictions of plume location and saturation distributions given the duration, rate, and depth interval of the CO_2 injection. These are very promising steps towards developing a more general purpose neural network model for plume prediction.

One of the challenges faced by neural network models is the ability to make predictions for situations outside of the training domain. Given the relatively large effort and computational time required to train a new model, this difficulty must be overcome. With this goal in mind, we demonstrate a transfer learning approach that can quickly be used to increase the ability of this network to make predictions outside the bounds of the initial training data. The experiments show that the transfer learning technique is very effective and efficient for new task domains that contain new spatial information (e.g. injecting into multiple depth intervals), while tasks that are associated with new temporal information require more data.

Migration of CO_2 in heterogeneous reservoirs is a complex multiphase flow problem subject to the competing and synergistic effects of gravitational, capillary and viscous forces. The excellent performance of the trained neural network model described here suggests that further development could result in practical and useful tools that could replace some of the tasks done by full physics simulators today. Nonetheless, the scope of this paper only considered the performance of a trained neural network for a small reservoir and cannot be directly applied to a real world problems. Limitations that need to be addressed in the future include: (1) the dimension of the system will significantly increase when dealing with a formation scale model and 3D problems; (2) training strategies for additional variables not considered here must be developed (e.g. salinity, formation temperature, porosity, permeability anisotropy, etc.); and (3) methods to account for post-injection migration of the CO_2 plume and the hysteretic formation of characteristics curves.

Data availability

The data set is generated with numerical simulator ECLIPSE 300. ECLIPSE license: E300 Version 2017.2. Build Number 30032000 (Arch: PC WINDOWS (64 bit, PC_IL13VS110_OPT)). The data set can be found at a public data repository hosted at Google drive: https://drive.google.com/drive/folders/1nHpXhKjOw_nBk8GvoYfzBj5tcBNaZ6Hh?usp=sharing. The Python codes of the neural network are released at https://github.com/gegewen/runet.

Authors' contribution

Gege Wen: conceptualization, methodology, software, validation, formal analysis, investigation, data curation, writing – original draft, writing – review & editing, visualization. Meng Tang: methodology, software, writing – original draft. Sally M. Benson: formal analysis, writing – review & editing, supervision, project administration, funding acquisition.

Declaration of Competing Interest

The authors report no declarations of interest.

Acknowledgments

We acknowledge the support from The Global Climate and Energy Project at Stanford University. We acknowledge the Center for Computational Earth & Environmental Sciences (CEES) at Stanford University for the computational resources. We would also like to acknowledge that the comments we received from the reviewers were extremely helpful and significantly improved this manuscript.

Appendix A. Comparison of plume behaviors at different injection rates

The behavior of a CO_2 plume is not only dependent on the injection volume, but also dependent on the injection rate. The example in Fig. A.1 shows two CO_2 plumes where the same volume is injected at 1.55 and 3.1 ton/day/m respectively. The permeability map and the injection perforation interval are identical for the two cases, yet, the plume shows different behavior. When injecting CO_2 at a lower rate for a longer duration, the plume in the first row of Fig. A.1 has higher CO_2 gas saturation near the reservoir top comparing to injecting at a larger rate in a shorter duration. This example also shows that the trained neural network model can successfully predict the behavior of each plume.

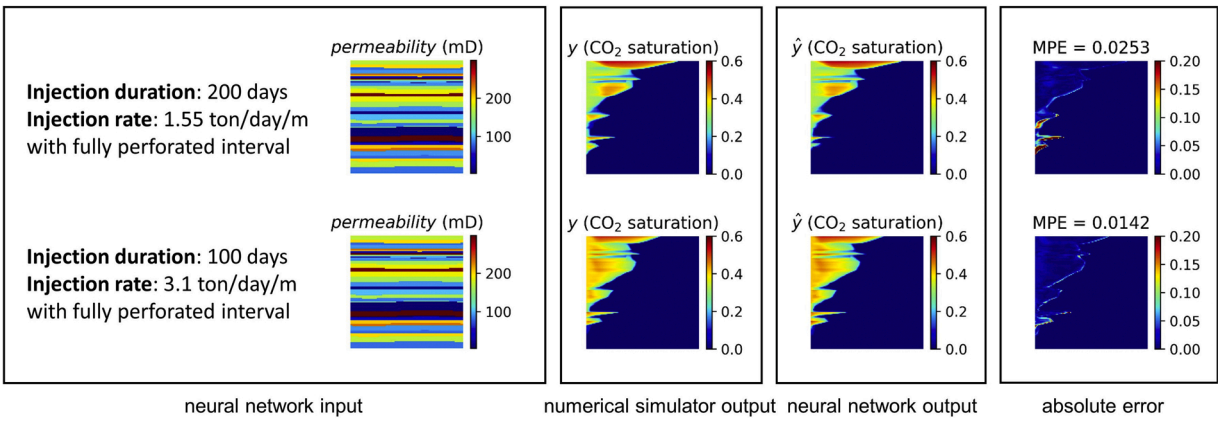

Fig. A.1. This figure shows two cases that have the same permeability map, perforation interval, injection volume, and different injection rates. The first column is the neural network input, the second column is the numerical simulator output, the third column is the neural network output, and the fourth column is the absolute error.

Appendix B. Hyperparameter selection

Learning rate determines the step size when minimizing for the loss function. In order to select the suitable learning rate for this problem, we compared a wide range of learning rates from 1e−2 to 1e−6. Fig. B.1a summarizes the evolution of the training and test RMSE with the number of epochs, for a range of learning rates. The model blows up when the learning rate is 1e−2 while the learning rates smaller than 1e−5 leads to very slow convergence. As a result, we selected 1e−4 as the optimal learning rate for this problem.

We also tested the model's performance at batch sizes of 8, 16, and 32. Batch size determines the number of training examples used in one iteration of gradient update. From Fig. B.1b, we can observe that the network proposed in this paper is almost insensitive to batch size. In this paper, we selected a batch size of 16 to avoid memory overflow.

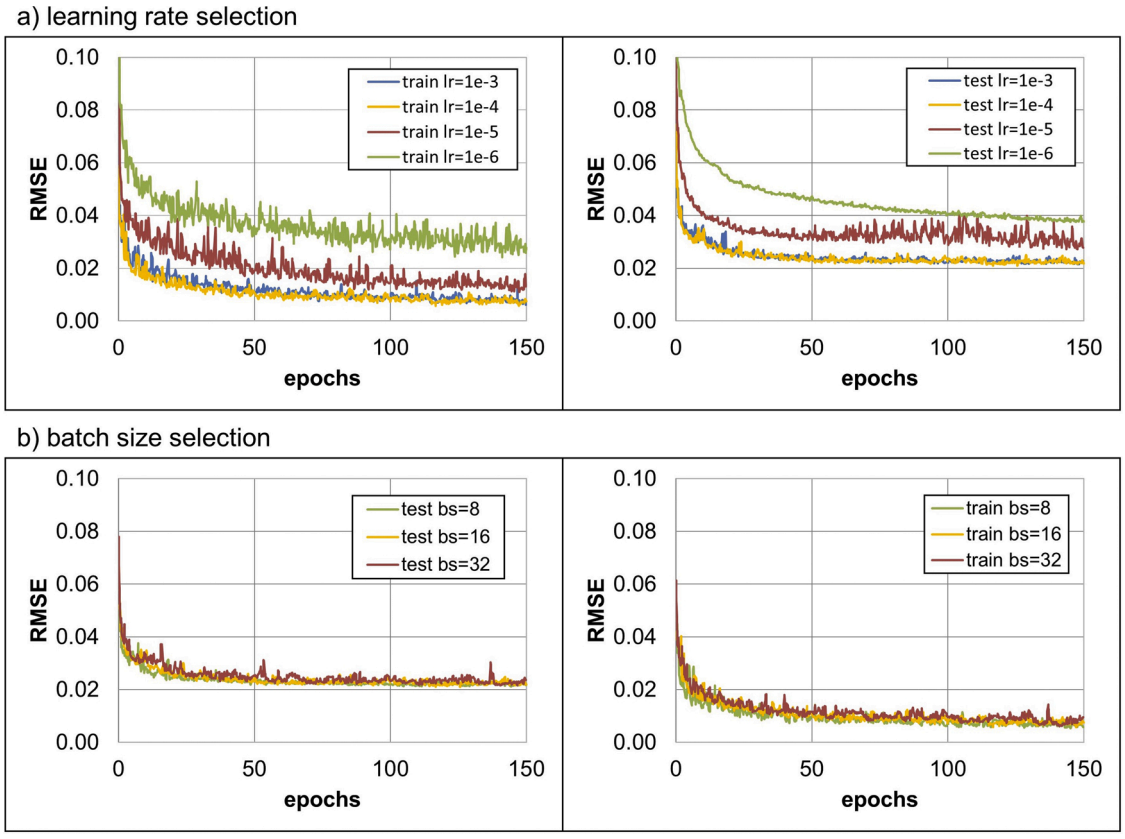

Fig. B.1. Training and test RMSE evolutions the number of epochs for (a) learning rate selection and (b) batch size selection. These models are all trained with the same training set which contains 6000 training samples.

Appendix C. Modified RU-Net architecture with pressure prediction

The details of the modified RU-Net architecture that is introduced in Section 6 are given in Table C.1.

Table C.1
Modified RU-Net architecture with pressure prediction. `Conv2D` denotes a 2D convolutional layer; `c` denotes the number of channels in the next layer; `k` denotes the kernel (or filter) size; and `s` denotes the size of the stride; `BN` denotes a batch normalization layer; `ReLu` denotes a rectified linear layer; `Conc` denotes a concatenation; `Unpool` denotes a unpooling layer; and `Padding` denotes a reflection padding layer.

Part	Layer	Output shape
Input		(64,64,2)
Encode 1	`Conv2D(c16k3s2)/BN/ReLu`	(32,32,16)
Encode 2	`Conv2D(c32k3s1)/BN/ReLu`	(32,32,32)
Encode 3	`Conv2D(c64k3s2)/BN/ReLu`	(16,16,64)
Encode 4	`Conv2D(c64k3s1)/BN/ReLu`	(16,16,64)
Encode 5	`Conv2D(c128k3s2)/BN/ReLu`	(8,8,128)
Encode 6	`Conv2D(c128k3s1)/BN/ReLu`	(8,8,128)
ResConv 1	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(8,8,128)
ResConv 2	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(8,8,128)
ResConv 3	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(8,8,128)
ResConv 4	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(8,8,128)
ResConv 5	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(8,8,128)
ResConv 6	`Conv2D(c128k3s1)/BN/ReLu/Conv2D (c128k3s1)/BN`	(8,8,128)
Decode 6	`Conc/Unpool(s2)/Padding/Conv2D (c128k3s1)/BN/Relu`	(8,8,128)
Decode 5	`Conc/Unpool(s1)/Padding/Conv2D (c128k3s1)/BN/Relu`	(16,16,128)
Decode 4	`Conc/Unpool(s2)/Padding/Conv2D (c64k3s1)/BN/Relu`	(16,16,64)
Decode 3	`Conc/Unpool(s1)/Padding/Conv2D (c64k3s1)/BN/Relu`	(32,32,64)
Decode 2	`Conc/Unpool(s2)/Padding/Conv2D (c32k3s1)/BN/Relu`	(32,32,32)
Decode 1	`Conc/Unpool(s1)/Padding/Conv2D (c16k3s1)/BN/Relu`	(64,64,16)
Output	`Conv2D(c2k3s1)`	(64,64,2)

Appendix D. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.ijggc.2020.103223.

References

Araya-Polo, M., Alpak, F.O., Hunter, S., Hofmann, R., Saxena, N., 2020. Deep learning-driven permeability estimation from 2D images. Comput. Geosci. 24, 571–580. https://doi.org/10.1007/s10596-019-09886-9.

Bazargan, H., Christie, M., Elsheikh, A.H., Ahmadi, M., 2015. Surrogate accelerated sampling of reservoir models with complex structures using sparse polynomial chaos expansion. Adv. Water Resour. 86, 385–399. https://doi.org/10.1016/j.advwatres.2015.09.009.

Cardoso, M.A., Durlofsky, L.J., Sarma, P., 2008. Development and application of reduced-order modeling procedures for subsurface flow simulation. Int. J. Numer. Methods Eng. https://doi.org/10.1002/nmee.2453.

Doughty, C., Pruess, K., 2004. Modeling supercritical carbon dioxide injection in heterogeneous porous media. Vadose Zone J. 847, 837–847.

Geneva, N., Zabaras, N., 2020. Modeling the dynamics of PDE systems with physics-constrained deep auto-regressive networks. J. Comput. Phys. 403, 109056. https://doi.org/10.1016/j.jcp.2019.109056.

Glorot, X., Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks. In: Teh, Y.W., Titterington, M. (Eds.), Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. PMLR, Chia Laguna Resort, Sardinia, Italy, pp. 249–256. In: http://proceedings.mlr.press/v9/glorot10a.html.

Hagan, M.T., Demuth, H.B., Beale, M.H., 1996. Neural Network Design. https://doi.org/10.1007/BF00738424.

Hamdi, H., Couckuyt, I., Sousa, M.C., Dhaene, T., 2017. Gaussian Processes for history-matching: application to an unconventional gas reservoir. Comput. Geosci. 21, 267–287. https://doi.org/10.1007/s10596-016-9611-2.

Haykin, S., 1999. Neural Networks and learning Machines, 2nd ed. doi: 978-0131471399, arXiv:1312.6199v4.

He, J., Sarma, P., Durlofsky, L.J., 2013. Reduced-order flow modeling and geological parameterization for ensemble-based data assimilation. Comput. Geosci. 55, 54–69. https://doi.org/10.1016/j.cageo.2012.03.027.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 770–778. https://doi.org/10.1109/CVPR.2016.90.

Ide, S.T., Jessen, K., Orr, F.M., 2007. Storage of CO2 in saline aquifers: effects of gravity, viscous, and capillary forces on amount and timing of trapping. Int. J. Greenhouse Gas Control 1, 481–491. https://doi.org/10.1016/S1750-5836(07)00091-6.

Ioffe, S., Szegedy, C., 2015. Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Bach, F., Blei, D. (Eds.), Proceedings of the 32nd International Conference on Machine Learning, PMLR, Lille, France, pp. 448–456. In: http://proceedings.mlr.press/v37/ioffe15.html.

IPCC, 2014. Climate change 2014: synthesis report – summary chapter for policymakers. Contribution of Working Groups I, II and III to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change.

Kingma, D., Ba, J., 2014. Adam: A Method for Stochastic Optimization. . Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015. arxiv:1412.6980.

Kitanidis, P.K., 2015. Persistent questions of heterogeneity, uncertainty, and scale in subsurface flow and transport. Water Resour. Res. 51, 5888–5904. https://doi.org/10.1002/2015WR017639.

Krevor, S., Blunt, M.J., Benson, S.M., Pentland, C.H., Reynolds, C., Al-Menhali, A., Niu, B., 2015. Capillary trapping for geologic carbon dioxide storage – from pore scale physics to field scale implications. Int. J. Greenhouse Gas Control 40, 221–237. https://doi.org/10.1016/j.ijggc.2015.04.006.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2012. Imagenet classification with deep convolutional neural networks. In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems 25. Curran Associates, Inc, pp. 1097–1105. http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Laloy, E., Hérault, R., Jacques, D., Linde, N., 2018. Training-image based geostatistical inversion using a spatial generative adversarial neural network. Water Resour. Res. 54, 381–406. https://doi.org/10.1002/2017WR022148.

Laloy, E., Hérault, R., Lee, J., Jacques, D., Linde, N., 2017. Inversion using a new low-dimensional representation of complex binary geological media based on a deep neural network. Adv. Water Resour. 110, 387–405. https://doi.org/10.1016/j.advwatres.2017.09.029.

Lee, H., Jang, Y., Kim, J., Jung, W., Sung, W., 2016. CO2 plume migration with gravitational, viscous, and Capillary forces in saline aquifers. Proceedings of the International Offshore and Polar Engineering Conference 2016-January 822–826.

Liu, Y., Sun, W., Durlofsky, L.J., 2018. A deep-learning-based geological parameterization for history matching complex models. Math. Geosci. https://doi.org/10.1007/s11004-019-09812-w.

MacMinn, C.W., Szulczewski, M.L., Juanes, R., 2011. CO2 migration in saline aquifers. Part 2. Capillary and solubility trapping. J. Fluid Mech. 688, 321–351. https://doi.org/10.1017/jfm.2011.379.

Mo, S., Zabaras, N., Shi, X., Wu, J., 2019a. Deep autoregressive neural networks for high-dimensional inverse problems in groundwater contaminant source identification. Water Resour. Res. 55, 3856–3881. https://doi.org/10.1029/2018WR024638.

Mo, S., Zabaras, N., Shi, X., Wu, J., 2020. Integration of adversarial autoencoders with residual dense convolutional networks for estimation of non-Gaussian hydraulic conductivities. Water Resour. Res. 56 https://doi.org/10.1029/2019WR026082.

Mo, S., Zhu, Y., Zabaras, N., Shi, X., Wu, J., 2019b. Deep convolutional encoder–decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media. Water Resour. Res. 55, 703–728. https://doi.org/10.1029/2018WR023528.

Nair, V., Hinton, G.E., 2010. Rectified linear units improve restricted Boltzmann machines. Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel 2010, 6421113 doi:10.1.1.165.6419.

NAS, 2018. Negative Emissions Technologies and Reliable Sequestration. https://doi.org/10.17226/25259.

NETL, 2017. Best Practices: Risk Management and Simulation for Geologic Storage Projects. https://www.netl.doe.gov/sites/default/files/2018-10/BPM_RiskAnalysisSimulation.pdf.

Nordbotten, J.M., Flemisch, B., Gasda, S.E., Nilsen, H.M., Fan, Y., Pickup, G.E., Wiese, B., Celia, M.A., Dahle, H.K., Eigestad, G.T., Pruess, K., 2012. Uncertainties in practical simulation of CO 2 storage. Int. J. Greenhouse Gas Control 9, 234–242. https://doi.org/10.1016/j.ijggc.2012.03.007.

Oliver, D.S., Chen, Y., 2011. Recent progress on reservoir history matching: a review. Comput. Geosci. 15, 185–221. https://doi.org/10.1007/s10596-010-9194-2.

Pruess, K., Nordbotten, J., 2011. Numerical simulation studies of the long-term evolution of a CO2 plume in a saline aquifer with a sloping caprock. Transp. Porous Media 135–151. https://doi.org/10.1007/s11242-011-9729-6.

Pruess, K., Oldenburg, C., Moridis, G., 1999. TOUGH2 User's Guide. Report LBNL-43134, p. 210.

Yang, Qiang, Pan, S.J., 2010. A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22, 1345–1359. https://doi.org/10.1109/TKDE.2009.191.

Radenovic, F., Tolias, G., Chum, O., 2018. Fine-tuning CNN image retrieval with no human annotation. IEEE Trans. Pattern Anal. Mach. Intell. 1–14. https://doi.org/10.1109/TPAMI.2018.2846566.

Reichstein, M., Camps-Valls, G., Stevens, B., Jung, M., Denzler, J., Carvalhais, Prabhat, N., 2019. Deep learning and process understanding for data-driven Earth system science. Nature. https://doi.org/10.1038/s41586-019-0912-1.

Remy, N., Boucher, A., Wu, J., 2009. Applied Geostatistics With SGEMS: A User's Guide.

Robert, C., Casella, G., 2000. Monte Carlo Statistical Methods. Springer Science & Business Media.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: convolutional networks for biomedical image segmentation. Medical Image Computing and Computer-Assisted Intervention (MICCAI) 234–241. Springer. http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a (Available on arXiv:1505.04597 [cs.CV]).

Rusk, N., 2015. Deep learning. Nat. Methods 13, 35. https://doi.org/10.1038/nmeth.3707.

Saadatpoor, E., Bryant, S.L., Sepehrnoori, K., 2010. New trapping mechanism in carbon sequestration. Transp. Porous Media 82, 3–17. https://doi.org/10.1007/s11242-009-9446-6.

Schlumberger, 2014. ECLIPSE Reference Manual.

Shen, C., 2018. A Transdisciplinary Review of deep learning research and its relevance for water resources scientists. Water Resour. Res. 54, 8558–8593. https://doi.org/10.1029/2018WR022643.

Song, Y., Sung, W., Jang, Y., Jung, W., 2020. Application of an artificial neural network in predicting the effectiveness of trapping mechanisms on CO2 sequestration in saline aquifers. Int. J. Greenhouse Gas Control 98, 103042. https://doi.org/10.1016/j.ijggc.2020.103042.

Strandli, C.W., Mehnert, E., Benson, S.M., 2014. CO2 plume tracking and history matching using multilevel pressure monitoring at the Illinois basin – Decatur project. Energy Proc. 63, 4473–4484. https://doi.org/10.1016/j.egypro.2014.11.483.

Szulczewski, M.L., MacMinn, C.W., Herzog, H.J., Juanes, R., 2012. Lifetime of carbon capture and storage as a climate-change mitigation technology. Proc. Natl. Acad. Sci. U.S.A. 109, 5185–5189. https://doi.org/10.1073/pnas.1115347109.

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.

Tartakovsky, D.M., 2013. Assessment and management of risk in subsurface hydrology: a review and perspective. Adv. Water Resour. 51, 247–260. https://doi.org/10.1016/j.advwatres.2012.04.007.

Tian, L., Wilkinson, R., Yang, Z., Power, H., Fagerlund, F., Niemi, A., 2017. Gaussian process emulators for quantifying uncertainty in CO2 spreading predictions in heterogeneous media. Comput. Geosci. 105, 113–119. https://doi.org/10.1016/j.cageo.2017.04.006.

Vandal, T., Kodra, E., Ganguly, S., Michaelis, A., Nemani, R., Ganguly, A.R., 2018. Generating high resolution climate change projections through single image super-resolution: an abridged version. IJCAI International Joint Conference on Artificial Intelligence 2018-July 5389–5393. arXiv:1703.03126v1.

Wen, G., Benson, S., 2019. CO2 plume migration and dissolution in layered reservoirs. Int. J. Greenhouse Gas Control 87, 66–79. https://linkinghub.elsevier.com/retrieve/pii/S1750583619300246.

Xiu, D., 2010. Numerical Methods for Stochastic Computations: A Spectral Method Approach. Princeton University Press.

Zhang, G., Lu, P., Zhu, C., 2014. Model predictions via history matching of CO2 plume migration at the Sleipner Project, Norwegian North Sea. Energy Proc. 63, 3000–3011. https://doi.org/10.1016/j.egypro.2014.11.323.

Zhong, Z., Sun, A.Y., Jeong, H., 2019. Predicting CO2 plume migration in heterogeneous formations using conditional deep convolutional generative adversarial network. Water Resour. Res. WR024592. https://doi.org/10.1029/2018WR024592.

Zhou, Z., Lin, Y., Zhang, Z., Wu, Y., Wang, Z., Dilmore, R., Guthrie, G., 2019. A data-driven CO2 leakage detection using seismic data and spatial-temporal densely connected convolutional neural networks. Int. J. Greenhouse Gas Control 90, 102790. https://doi.org/10.1016/j.ijggc.2019.102790.

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.

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. https://doi.org/10.1016/j.jcp.2019.05.024.