Petroleum Science 21 (2024) 716–728

Contents lists available at ScienceDirect

Petroleum Science

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

Original Paper

Better use of experience from other reservoirs for accurate production forecasting by learn-to-learn method

Hao-Chen Wang [a, h], Kai Zhang [a, c, *], Nancy Chen [b, **], Wen-Sheng Zhou [d, e], Chen Liu [d, e], Ji-Fu Wang [f], Li-Ming Zhang [a], Zhi-Gang Yu [g], Shi-Ti Cui [f], Mei-Chun Yang [f]

[a] China University of Petroleum (East China), 66 Changjiang West Road, Qingdao West Coast New Area, Qingdao, 266580, Shandong, China
[b] University of Calgary, 2500 University Dr NW, Calgary, T2N 1N4, Alberta, Canada
[c] Qingdao University of Technology, Qingdao, 266071, Shandong, China
[d] State Key Laboratory of Offshore Oil Exploitation, Beijing, 100028, China
[e] CNOOC Research Institute Ltd., Beijing, 100028, China
[f] CNPC Tarim Oilfield Branch Company, Korla, 841000, Xinjiang, China
[g] National Engineering Laboratory for Exploration and Development of Low-Permeability Oil and Gas Fields, Xi'an, 710000, Shaanxi, China
[h] Sinopec Matrix Co., LTD, Qingdao, 266000, Shandong, China

ARTICLE INFO

Article history:
Received 15 September 2022
Received in revised form
14 April 2023
Accepted 17 April 2023
Available online 8 May 2023

Edited by Jia-Jia Fei

Keywords:
Production forecasting
Multiple patterns
Few-shot learning
Transfer learning

ABSTRACT

To assess whether a development strategy will be profitable enough, production forecasting is a crucial and difficult step in the process. The development history of other reservoirs in the same class tends to be studied to make predictions accurate. However, the permeability field, well patterns, and development regime must all be similar for two reservoirs to be considered in the same class. This results in very few available experiences from other reservoirs even though there is a lot of historical information on numerous reservoirs because it is difficult to find such similar reservoirs. This paper proposes a learn-to-learn method, which can better utilize a vast amount of historical data from various reservoirs. Intuitively, the proposed method first learns how to learn samples before directly learning rules in samples. Technically, by utilizing gradients from networks with independent parameters and copied structure in each class of reservoirs, the proposed network obtains the optimal shared initial parameters which are regarded as transferable information across different classes. Based on that, the network is able to predict future production indices for the target reservoir by only training with very limited samples collected from reservoirs in the same class. Two cases further demonstrate its superiority in accuracy to other widely-used network methods.

1. Introduction

Oilfield development and production forecasting play a vital guiding role in the future planning of oilfields. The traditional method used to forecast consists of the most traditional analytical method and the numerical simulation method. The most classical one is the decline curve analysis method (DCA). Hakim Elahi (2019) uses DCA and updates its parameters at the time that operation on wells changes, but they have to adjust parameters manually. To make the DCA more automatic, Masini et al. (2020) and Li et al. (2022) take advantage of machine learning and the Bayesian method to discard the outliers and generate the distribution of parameters in DCA to make the curve more flexible. When facing a multiphase flow situation, a single curve is not expressible enough to describe the rule. As a result, Makinde and Lee (2016) use hybrid (combination) decline curve analysis methods to reflect multiphase flow. However, it is still aiming at predicting trend information though the decline is fast to calculate. Another way to perform production forecasting is through numerical simulation, which solves the equations governing fluid flow in porous media numerically, i.e., through a series of iterations. Nolen (1973) and Douglas et al. (1997) proposed numerous numerical simulators that represent underground flow processes. To address the time-consuming nature of numerical simulators, Bakhty et al. (2020) proposed simplified formulae that are based on material balance to

* Corresponding author.
** Corresponding author.
E-mail addresses: zhangkai@upc.edu.cn (K. Zhang), snchen@ucalgary.ca (N. Chen).

https://doi.org/10.1016/j.petsci.2023.04.015

save time at the expense of accuracy. To deal with a more complex recovery method such as steam-assisted gravity drainage, Dehdari and Deutsch (2020) added Butler's Original Theory into the original method to extend its functionality. This type of method (Zhong and Wu, 2017) requires a deep understanding and accurate formulae construction of the mechanism which is too difficult to implement when encountering unconventional developing policies or reservoirs.

Another approach to tackle the production forecasting problem is machine learning methods. It completes the training of internal parameters through a large number of samples (Song et al., 2017) to realize the prediction of future data (Oreshkin et al., 2021). The most commonly used network (Li et al., 2019) structure for time series data is Recurrent Neural Network (RNN). Li and Han (2017) predict the relation between production rate and fracture parameters in unconventional reservoirs with an RNN network. Alimohammadi et al. (2020) adopt a revised recurrent neural network (RNN) based network to perform short-term prediction of oil production rate. To show its advantages, Temizel et al. (2020) compare the differences between DCA, XGboost, and another RNN network named long short-term memory (LSTM), and show that LSTM works best and this performance is also proved by Gupta et al. (2021). To extend its application to gas indices prediction, Kocoglu et al. (2021) utilize an upgraded RNN network named Bidirectional Long Short Term Memory (Bi-LSTM). Still, these proposed networks cannot make use of the static information of wells, such as permeability or porous. Having obtained the geological model Du et al. (2022) made an effort to add information from the geological model into history that was learned by the network in the training process. Precisely, they merge the convolutional layers into LSTM to distill temporal and spatial features simultaneously. However, Razak et al. (2021) resort to existing formulae by using the sum of results from a network and numerical simulator, thus improving accuracy in trend prediction at the expense of modeling time. However, an existing limitation is the insufficient and unbalanced samples. In order to make better use of the huge amount of data that has distribution differences from the target data, researchers begin to study transfer efficiency. Researchers have successfully employed transfer learning to predict many goals. Odi et al. (2021) use transfer learning to help fracture matching. Dong et al. (2021) make use of stacked LSTM by transferring their parameters from a well-trained network in a public dataset. Mohd Razak et al. (Mohd et al., 2022) adopt this method in unconventional reservoirs by taking the advantage of public datasets in the same transfer way. The aforementioned transfer concept is to copy parameters from a well-trained network in a public dataset at the initial stage of network finetuning. This operation only works well under the assumption that the target data distribution does not deviate much from the public data distribution. Otherwise, the source and target may distract each other leading to chaotic results. However, the public dataset may include production indices from many different classes of reservoirs that have few samples respectively, and their data are much different from each other.

In order to solve the above problems, this paper proposes an RNN-based model-agnostic machine learning (RMAML) framework which derives from a few-shot learning method by (Iwata and Kumagai, 2020). Currently, combining few-shot learning with RNN networks is very rare, let alone the network designed for reservoir data. This is because few-shot learning was not originally designed for sequences. The term "model-agnostic" refers to a particular computing framework, often known as a "meta-learner". Most deep learning network models can be integrated with this meta-learner as a base learner. Technically, it is able to not only extract common laws but also effectively reduce mutual interference between the samples that are largely different, and thus it has a stronger transferability among other reservoirs and target reservoirs.

2. Problem statement

In this section, we hope to state our problem more clearly. Different classes of reservoirs, imply different permeability fields and different developing policies, such as different control rates in production and injection and different well placement, and thus may result in a large difference in the distribution of their production indices. However, this big difference in distribution may lead to mutual distraction during training. In consequence, the parameters in the network can hardly find their way to optimal points.

Here, we use simulated data by Eclipse software (Schlumberger) on a self-defined square to describe the problem, as shown in Fig. 1. In practice, we take permeability fields, well placement, and control regimes as variables to construct models. In each model, we generate the indices sequences, including liquid production rate, injection rate, oil and water production rate, water injection rate, and other accumulative quantities. Then we use the principal component analysis (PCA) method (Jolliffe et al., 2016) to turn sequences in each model into two-dimensional data for easier display, as shown in Eqs. (1) and (2).

The PCA is to find a new basis coordinate that maximizes the variance of the data matrix features and minimizes the covariance between features. Here X is the 0−1 normalized input matrix which has the shape of m by n where m represents the number of samples and n denotes the number of features. C is the covariance matrix of X, $a_1, a_2, \ldots, a_n$ are the eigenvectors corresponding to the first n largest eigenvalues of matrix C.

$$C = \frac{XX^T}{m} \tag{1}$$

$$Y = [a_1, a_2, \ldots, a_n]^T X \tag{2}$$

The distribution of data after PCA is shown in Fig. 2, where each point is a production sequence and points in different colors are from reservoirs with different variables. For instance, in Fig. 2(a) and (b), different color means the sequence is generated from different permeability field while other variables are the same. We can find that when the permeability field and control regimes changed, the data remained fairly evenly distributed. But when the well location changed, the data began to show distinct distribution clusters for each class. Even worse, the distance between clusters is much larger than that of points in a single cluster. The performance in prediction by the common method is shown in the Case Study section. Therefore, we can learn that different classes of reservoirs that have different well placements may lead to entirely unrelated data representation and thus it is hard to infer the future performance of one from others.

3. Methodology

3.1. Related work

In this section, we propose an RNN-based model-agnostic machine learning (RMAML) method for multi-classes prediction, referring to few-shot learning and meta-learning methods (Lemke and Gabrys, 2010; Oreshkin et al., 2021; Caena and Stringher, 2022) which are widely used in the field of few-shot tasks. The main methods of few-shot learning can be roughly divided into two categories, gradient-based methods, and conditional-based methods. The gradient methods (Yang et al., 2021) achieve

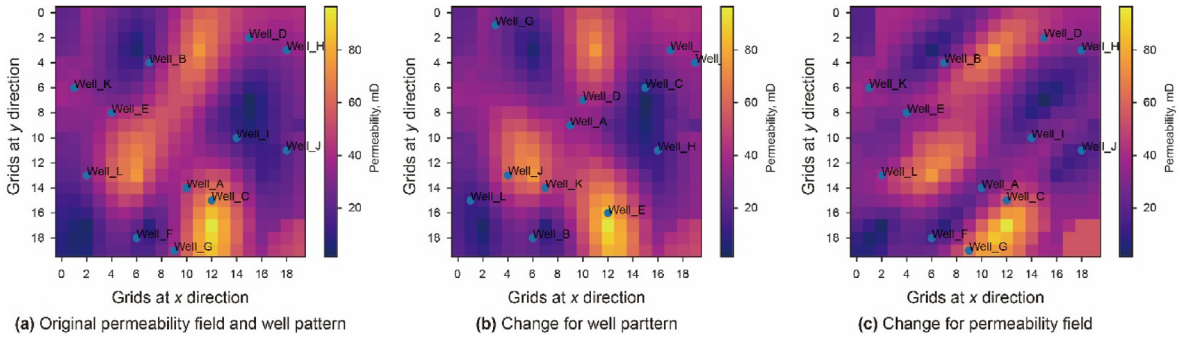

Fig. 1. Exhibition of different permeability and well pattern.

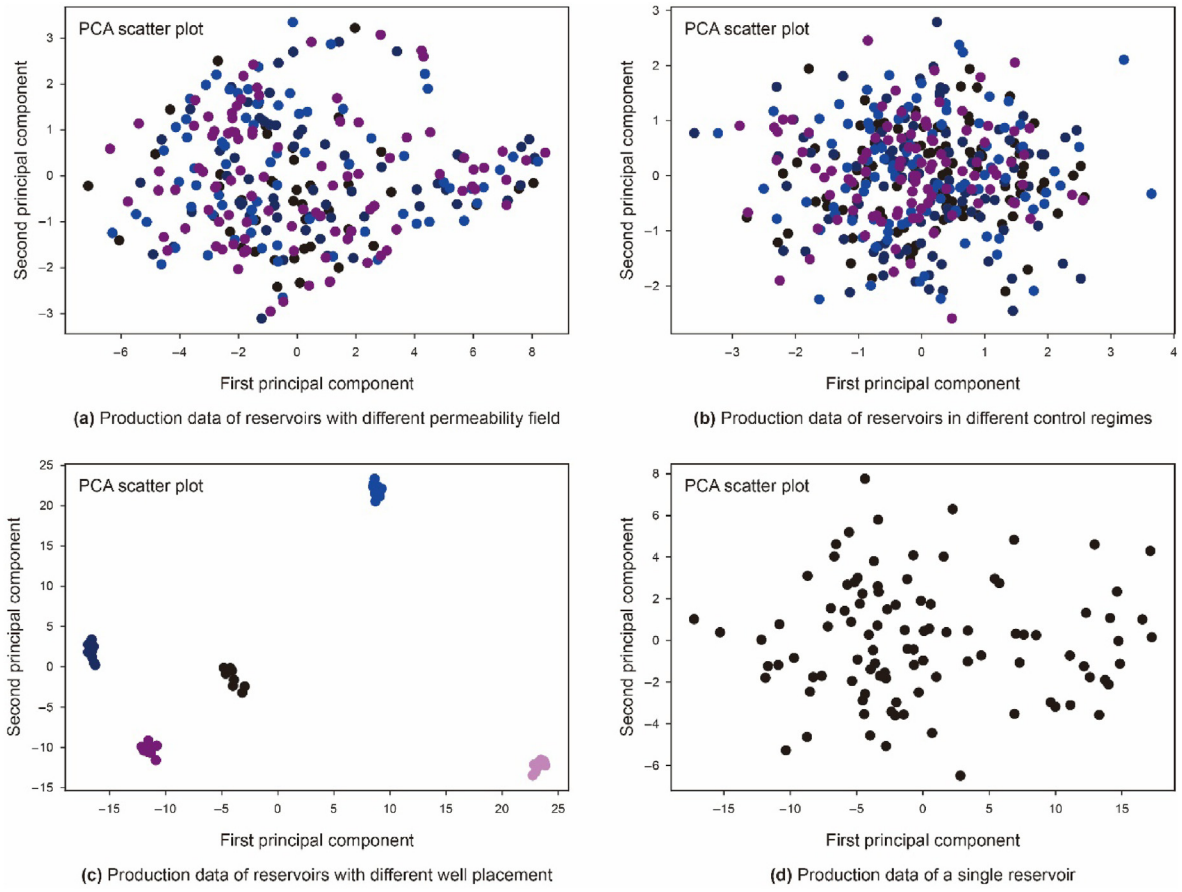

Fig. 2. The distribution of data when changing different factors.

optimizing the classification model independently in different tasks and update the common initial parameters. Since various tasks use updated parameters in their own way, the differences between tasks are considered. In order to improve the calculation efficiency, Simon et al. (2020) propose a first-order gradient update method to improve training speed. Ma et al. (2022) also proposed a multi-model version to enrich the calculation method for initial parameters and make it more flexible. Another type of method is the conditional neural network (CLNN) method (Wang et al., 2019), which is also within the concept of the learn-to-learn approach. But different from gradient methods, it uses sample information as context information, which is fused into the input information of the target after transformation. Since the method is affected by the relationship between input information and label information, Lu et al. (2022) propose a series of regularization technique to strengthen the influence of the context part on the results. In general, the gradient method is more versatile and more suitable for few-shot training, while the conditional method is more flexible and has a stronger fitting ability, but requires more samples to assist learning (Gao et al., 2022).

3.2. Learn-to-learn concept and problem symbolization

In terms of training strategy, different from general network training, we use the learn-to-learn concept to train networks. Intuitively, we are trying to force the network to learn the rules from samples in their own cluster but share the way of learning between clusters. In practice, we carry out this idea by learning the

optimal initial parameters of the network for all clusters and independently updating them in each cluster without interfering with others.

Technically, the points in different clusters represent different classes of reservoirs that have different control regimes or permeability fields, or well placement. Then we construct a self-defined recurrent neural network, denoted function f with initialized parameters θ optimized by the MAML framework, that can quickly learn the map from both control regimes D and static attributes of wells S to oil production rate Y_{test} for a target reservoir with a small number of samples. Here we denote $Y_{\text{test}} = f(X_{\text{test}})$ where $X_{\text{test}} = D, S,$ and the set includes both X and Y which are named query data $\mathscr{Q}_{\text{test}} = X_{\text{test}}, Y_{\text{test}}$. Other sets in the same cluster with $\mathscr{Q}_{\text{test}}$ are called support set $\mathscr{S}_{\text{test}} = X_{\text{test}}, Y_{\text{test}}$ representing the information from similar classes of reservoirs. In order to find the best initial parameters θ for f, the network has to learn from a large number of known pairs $\mathscr{S}^{\tau_i}_{\text{train}}$, $\mathscr{Q}^{\tau_i}_{\text{train}}$ from a different cluster i. Generally, we regard prediction in different clusters as different tasks $\tau = \tau_1, \tau_2 \ldots, \tau_3$ where each of τ_i denotes a certain cluster. Both $\mathscr{S}^{\tau_i}_{\text{train}}$ and $\mathscr{Q}^{\tau_i}_{\text{train}}$ are from the same cluster i and $\mathscr{S}_{\text{train}}$ and $\mathscr{Q}_{\text{train}}$ construct training sets. After training, the parameters in the network are updated to optimal θ'. Then it is further trained through the support set $\mathscr{S}_{\text{test}}$ in target cluster to $\tilde{\theta}_t$ which will be used to predict the future of the target reservoir $\mathscr{Q}_{\text{test}}$ in the test set. In the subsequent algorithm description, $\mathscr{S}$ and $\mathscr{Q}$ are used to represent the support set and query set respectively in order to simplify symbol expression in both the training and testing process. In training, $\mathscr{S}$ and $\mathscr{Q}$ refer to $\mathscr{S}_{\text{train}}$ and $\mathscr{Q}_{\text{train}}$, while during testing, $\mathscr{S}$ and $\mathscr{Q}$ mean $\mathscr{S}_{\text{test}}$ and $\mathscr{Q}_{\text{test}}$.

3.3. Workflow

Our method deals with data in four steps. The first step is data collection which splits data recorded during reservoir development into samples. Then, the feature normalization step transforms data into a standard scale ranging from 0 to 1. After that, the network propagation updates the parameters in the network through the training dataset and predicts the future production index of the target reservoir. For measuring the accuracy of our work and calculating the gradient for backpropagation, we have a metric selection operation at the end of the process.

3.3.1. Data collection

We collect samples from development experience by different well patterns. The input data is divided into two parts: static data and dynamic data. Static data is the attributes of wells, including location information, permeability information, and porosity information. Dynamic data includes control regimes for well production. The target output is the oil production rate for each well.

We classify them into different reservoir classes according to clusters shown in Fig. 3. All series in each cluster are divided into two sets named support set and query set in proportion 0.7 and 0.3, respectively. These two data sets' names are taken from the few-shot learning concept. The support set and query set are utilized for inner loop training and outer loop training, respectively and there is no intersection between them. Then, each sample is constructed with a fixed proportion of data sampled from two sets. It is worth noticing that the data of the target reservoir for prediction must be in the query set.

This type of sample construction is determined by the nature of the method. Because our method aims at finding out the shared information between different clusters while preserving the differences between them. Therefore, the support set shows the network the distinct features of each cluster, while the query set enables the network to find out the information that can be generalized with the help of previous hints of their differences.

3.3.2. Classification

Since the sample data comes from different well pattern forms and different permeability field distributions, the distribution pattern of the final production data is too scattered. For network training, a really preferable data distribution should be relatively uniform. Therefore, we try to classify these scattered data so that the data presents a relatively uniform distribution within its own class. In this paper, we use the methods of PCA and k-means to achieve data classification, and the effect after classification is shown in Fig. 2.

3.3.3. Feature normalization

In terms of data normalization, we use Min-Max standardization to scale the data into the range 0–1, as Eq. (3), and all features use their respective maximum and minimum values as parameters.

$$\tilde{x}^i = \frac{x^i - x^i_{\min}}{x^i_{\max} - x^i_{\min}} \tag{3}$$

where i is the index of features, $\underset{\sim}{x}^i$ is the normalized data, x^i is the raw data, $x^i_{\min}$ is the minimum value and $x^i_{\max}$ is the maximum value of i^{th} feature.

3.3.4. Performance metrics

The absolute error L_1 is used as our loss function in both the inner and outer loops. It measures the absolute error between the real results and predicted values, as shown in Eq. (4). Here n is the total number of samples, y_i and y_i are the target results and the corresponding prediction value in the grid block, respectively. It is worth mentioning that in the training process, the injection and production rate have a strong decisive role in the actual results. Therefore, in order to avoid the network ignoring context information, we utilize Kullback-Leibler divergence (D_{KL}), as shown in Eq. (5), as an extra regularization term (Li et al., 2022), where $P(x)$ and $Q(x)$ are the probability distribution of x.

$$L_1 = \sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{4}$$

$$D_{\text{KL}}(P|Q) = \sum_{x \in X} P(x) \log\left(\frac{P(x)}{Q(x)}\right) \tag{5}$$

3.3.5. Network training

Our framework can be divided into two parts, named the inner part and the outer part. The inner part is to update the network for each cluster or task separately and the outer part is to search for a common initial point. The whole algorithm process is shown in Table 1. For more specific theoretical proof, please refer to (Yang et al., 2021).

3.3.5.1. Inner loop.

The inner loop network is proposed by ourselves and used for the separate training of every single task or cluster. When the training of each task is completed, the network parameters are updated from the same initial value θ, but converge to different optimal values $\tilde{\theta}_\tau$ for each task τ, as shown in Eq. (6).

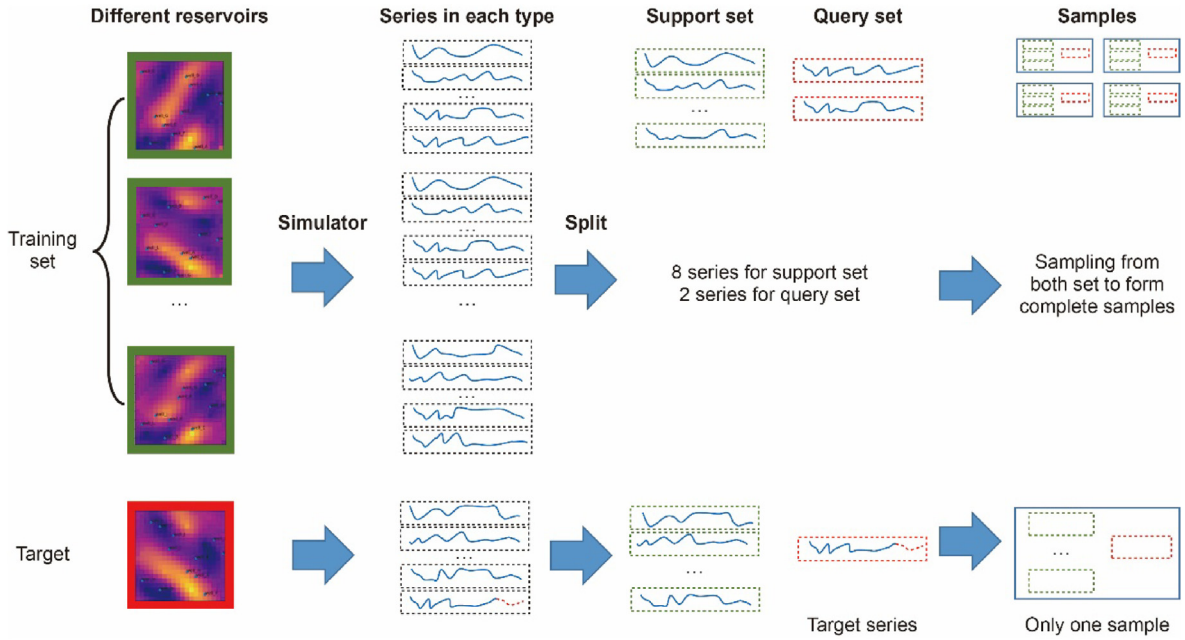

Fig. 3. The process of constructing samples for both training and target prediction.

Table 1
Algorithm training Workflow.

Algorithm: RMAML for production forecasting

Require: $p(\tau)$: distribution over tasks

Require: α, β: learning rate of parameters in the inner loop and outer loop

Randomly initialize: θ

While not done do (outer loop):

Sample batch of tasks $\tau_i \sim p(\tau)$

for all τ_i do: (inner loop):

Sample K datapoints $\mathcal{D}_s^i = \{x_s^i, y_s^i\}$ from $\mathcal{S}_{\text{train}}$ in τ_i

Evaluate $\nabla_{\theta \mathcal{L}_{\mathcal{D}_i}}(f_\theta)$ using $\mathcal{D}_s^i$ and $\mathcal{L}_{\tau,\mathcal{S}}$ in Eq. (4) and Eq. (5)

Compute adapted parameters with gradient descent: $\tilde{\theta} = \theta - \alpha \nabla_{\theta \mathcal{L}_{\tau,\mathcal{S}}}(f_\theta)$

Sample data points $\mathcal{D}_q^i = \{x_q^i, y_q^i\}$, from $\mathcal{Q}_{\text{train}}$ in τ_i for the meta-update

end for

Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\tau_i \sim p(\tau)} \mathcal{L}_{\tau,\mathcal{Q}}(f_{\tilde{\theta}})$ using $\mathcal{D}_q^i$ and τ_i in Eq. (4) and Eq. (5)

End while

$$\tilde{\theta}_\tau = \text{minimum}_\theta \mathscr{L}_{\tau,S}\left(f_{\tau,S}^k(\theta)\right) \tag{6}$$

where $f_{\tau,S}^k$ is the operator that updates θ by k iterations using data about task τ sampled from $\mathscr{S}$. In few-shot learning, $\mathscr{L}$ corresponds to a loss function that performs gradient descent on batches of data sampled from τ.

The structure of the network is shown in Fig. 4 where C in a circle represents concatenate operation. To calculate the impact of historical information, we adopt the RNN framework as its main feature extraction structure. Specifically, we use two different GRU networks as a base learner to extract features for static features and dynamic

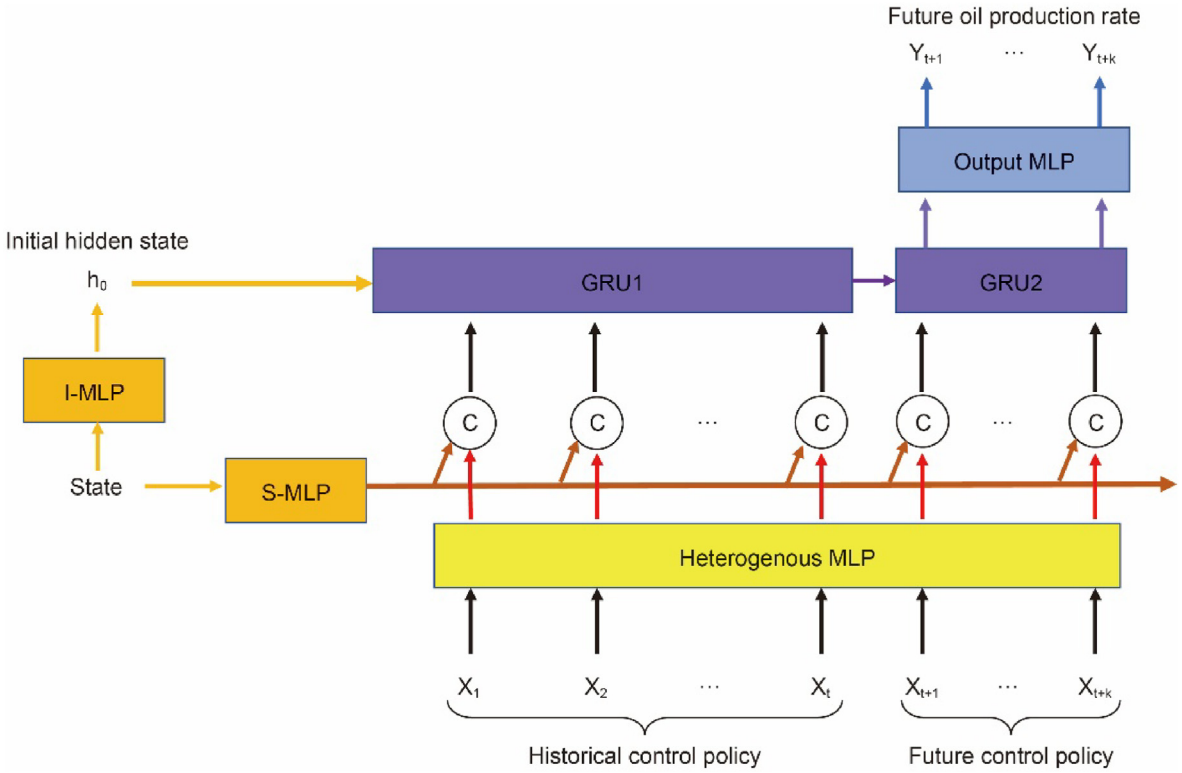

Fig. 4. The structure of the inner-loop network.

features named static network and dynamic network. Each GRU network is composed of four equations, as shown by Eqs. (7)–(10).

$$r_t = \sigma\left(W^r * H_t + U^r * HL_{t-1}\right) \tag{7}$$

$$i_t = \sigma\left(W^i * H_t + r_t \odot U^i * HL_{t-1}\right) \tag{8}$$

$$z_t = \sigma\left(W^z H_t + U^z HL_{t-1}\right) \tag{9}$$

$$HL_t = z_t \otimes HL_{t-1} + (1 - z_t) \otimes i_t \tag{10}$$

Here, $\odot$ is the matrix dot product and $\otimes$ is the element-wise product. Reset Gate r_t determines the degree of change in hidden features. σ is an active function and matrices W and U are weight parameters; H_t is the input hidden feature at the current step, and H^L_{t-1} is a long-term feature left from history. i_t is the updated hidden feature at the current step by considering historical features. z_t is the update gate, which determines the change degree of origin features affected by fusion features, and it is also a weight parameter. H^L_t is the final updated specific features of each field.

Furthermore, combined with the future control regimes, the network produces a forecast of future oil production rates. In addition, we also employ a learnable multi-dimensional adjacency matrix to simulate inter-well interference. Since this matrix is entirely learned by the network without any physical information, we cannot figure its physical meaning out yet. It is worth mentioning that, in order to distinguish different flow phases in control regimes, we divide water injection and liquid production into two different feature dimensions. Further, we directly adopt the idea of the heterogeneous graph, using two different multi-layer perception networks (MLP), as shown in Eq. (11) for oil and water wells to map two types of features into a commonly hidden feature space. W_i and b_i are the weight matrix and bias matrix, respectively. σ is the activation function.

$$y = \sigma(W_1 \odot (\sigma(W_2 \odot X + b_1) + b_2)) \tag{11}$$

3.3.5.2. Outer loop.

The outer loop is a core part of this work looking for the shared optimal initial parameters of the network. At first, the inner loop starts from the initial parameters θ and updates them to $\tilde{\theta}$ according to their own gradient based on support sets $\mathscr{S}$ in each task. Then the outer loop further optimizes the original parameters θ with the gradient calculated from updated parameters $\tilde{\theta}$ on query dataset $\mathscr{Q}$, as shown in Eq. (12), so that each task shared the same optimal initial parameter θ.

$$\theta' = \text{minimum}_\theta \mathbb{E}_\tau\left[\mathscr{L}_{\tau,\mathscr{Q}}(\tilde{\theta}_\tau)\right] = \text{minimum}_\theta \mathbb{E}_\tau\left[\mathscr{L}_{\tau,\mathscr{Q}}\left(f_{\tau,\mathscr{S}}(\theta)\right)\right] \tag{12}$$

In gradient calculation, we first record the process of each update in the inner loop according to support set $\mathscr{S}$. This updating process is also a non-in-place operation, so it also generates gradients on initial parameter values θ as well. Then, we use the query set $\mathscr{Q}$ to calculate the loss in each task (except in the test part, because the query set in the test part is the target data that need to be predicted) thus the second-order gradient is obtained, as shown in Eq. (13). Here, $\tilde{\theta} = f_{\tau,\mathscr{S}}(\theta)$, and $f'_{\tau,\mathscr{S}}(\theta)$ is the Jacobian matrix of the update operation. $f_{\tau,\mathscr{S}}, f_{\tau,\mathscr{Q}}$ corresponds to adding a sequence of gradient vectors to the initial vector, i.e. $f_{\tau,\mathscr{S}}(\theta) = \theta + g_1 + g_2 + \ldots + g_k$. Finally, a back-propagation comes to work to update θ. The update process is shown in Fig. 5 where the blue curve arrow is the updated trajectory of each inner loop and the colorful arrows are

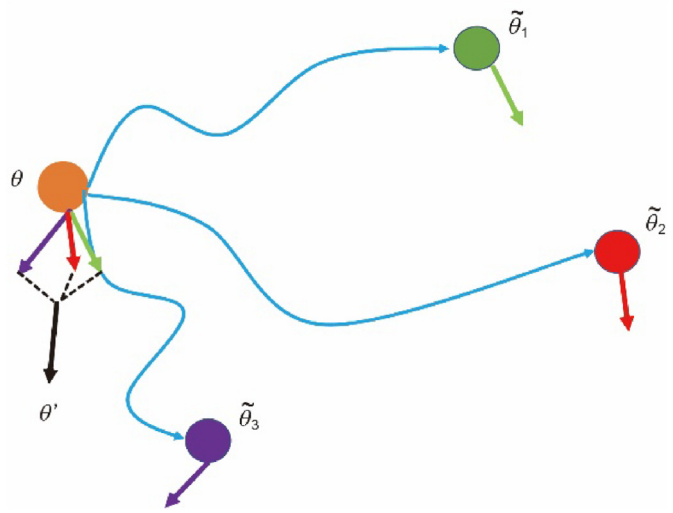

Fig. 5. The update process for initial parameters θ.

the gradient of clusters in the outer loop. Furthermore, the sum of colorful arrows (black arrow) is the final gradient of initial parameters θ for one update step.

$$g = \frac{\partial}{\partial \theta} \mathscr{L}_{\tau,\mathscr{C}}\left(f_{\tau,\mathscr{S}}(\theta)\right) = f'_{\tau,\mathscr{S}}(\theta)\mathscr{L}'_{\tau,\mathscr{C}}(\tilde{\theta}) \tag{13}$$

3.3.6. Prediction

The optimal initial parameters θ' of the network after training can then be used for production prediction of the target type of reservoir. Specifically, we use the network with parameter θ' and the support set $\mathscr{S}_{\text{test}}$ of the target type for further training to obtain the network with parameter $\tilde{\theta}_t$ which is applicable to the target reservoir $\mathscr{C}_{\text{test}}$ Finally, the network predicts future oil production rate based on the historical data and control policy of the target reservoir.

4. Case Study

4.1. Case1: square model

The self-designed square model is primarily used to confirm our approach is available. Each grid in the model has a grid number of 20*20*1 and a size of 50 m*50 m*10 m.

4.1.1. Dataset setup

In this part, we use a dataset generated by numerical simulation to validate the effectiveness of the proposed method. To obtain the dataset in the distribution of multiple clusters as shown in the problem statement. We have run simulations on various classes of reservoirs by the software Eclipse (Schlumberger). In practice, we have designed 50 classes of reservoirs with different permeability field control regimes and well placement, as shown in Fig. 6 which exhibits 4 different reservoir classes. Each of them runs 10 times numerical simulations to generate 10 series and each series contains the production rate of wells in this class of reservoir. Precisely, each series contains 50 time steps and each step length is 30 days. Then, we use the data from the first 49 reservoirs to form the training part and the last one as the target class which is also considered a test set. More precisely, for each class, we split the series into the support sets and the query sets with 8 series and 2 series respectively. It is worth noticing that the future part of the query set in the target class is what we are aiming for, we thus draw it with the red dash line. Then we randomly sampled 3 series from the support set and 1 series from the query set in the same class to construct every sample.

Specifically, 12 wells including 8 production wells and 4 injection wells are located randomly in each geological model. Based on their random attributes, we used the indicated kriging interpolation to interpolate the permeability field and porosity field with random azimuth and lag distance. Since these wells are relatively far apart, we have limited their nugget values to fixed values to improve stability and visuals. In the process of generating the production sequence, we make use of stochastic control regimes to control production and injection. The liquid production rate and water injection rate range from 0 to 100 m^3/d.

In practice, the input of the network contains two parts, that is, the dynamic part and the static part. We consider control regimes as the dynamic part and permeability as the static part then output oil production rate. The reason that we do not input the location of wells is that we treat them implicitly as the criteria for discriminating tasks. Here, we utilize L_1 to present $L_1 + D_{\text{KL}}$ in our paper to measure the accuracy of our work.

4.1.2. Performance

We find that both the training losses and testing loss can converge to a very small level, as shown in Fig. 7 and Table 2. Besides, we can clearly see that the network can achieve an accurate prediction on the target sequence after being further trained by 9 samples of the same class in the support set, as shown in Fig. 8 where the yellow line is the prediction result of our method and green points are real production rate.

Technically, RMAML is a meta-learning method that can converge parameters into a class-adapted optimal point rather than the common optimal point calculated by general one-order network work such as MLP. Precisely, the parameters in the network will be independently further updated when encountering any support set after having optimized its initial value by the RMAML method. Meantime, the RNN class network in the inner loop has a powerful ability to match the relation between input and output. Therefore, a nice performance is achieved.

4.1.3. Comparison with other methods

We have introduced many network methods in the Introduction section. Many of them have achieved nice performance, but some instinct limitation is still hidden in their methods. To make it more clear, we compare the results from the other three common methods. The first one is transfer learning, that is, regardless of reservoir class, all classes except the target class are fused and trained in mini-batch, then finetuning the learned parameters with few samples in the target class. Another is the class-only method, that is, only the first 9 samples of the target pattern are used, completely blocking the interference from other classes. The third one is mixed training, which combines samples from all classes of reservoirs including the samples of support set in the test part and tests the network on the query set of the test part. The test loss values of the four methods are shown in Table 2 and the final results are shown in Fig. 8. It is obvious that our method performs best among those methods. Specifically, the largest error is the result predicted by the mixed training method. When we analyze it from the perspective of data distribution, this is due to the centers of other clusters being far away from that of the target cluster. Therefore, when the network calculates the joint distribution of (x, y), it will be trying its best to cover all clusters, and in consequence, the target cluster will not be valued. For the class-only method, since the distribution of 10 samples inside the target class is not

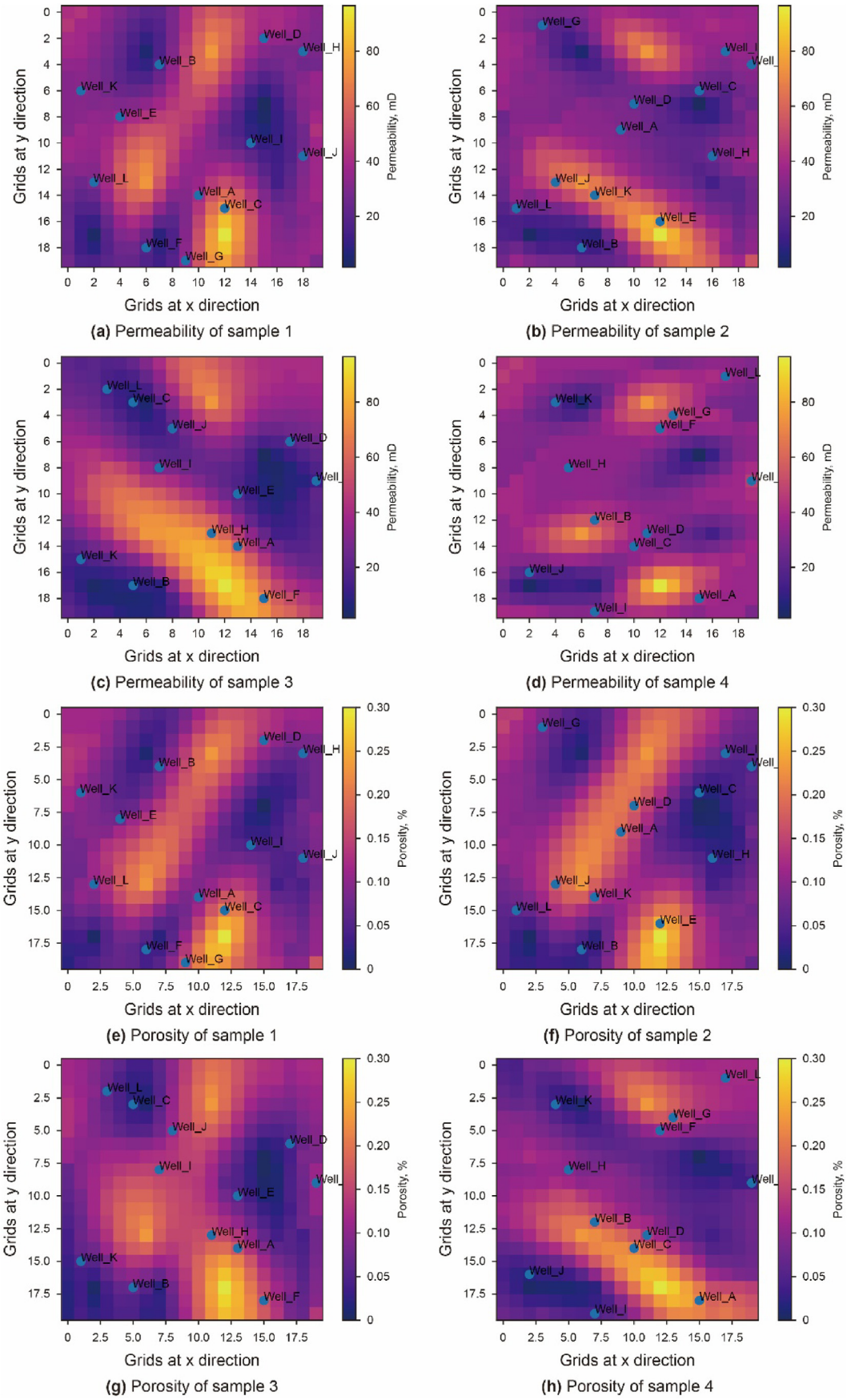

Fig. 6. Random permeability and porosity field in samples.

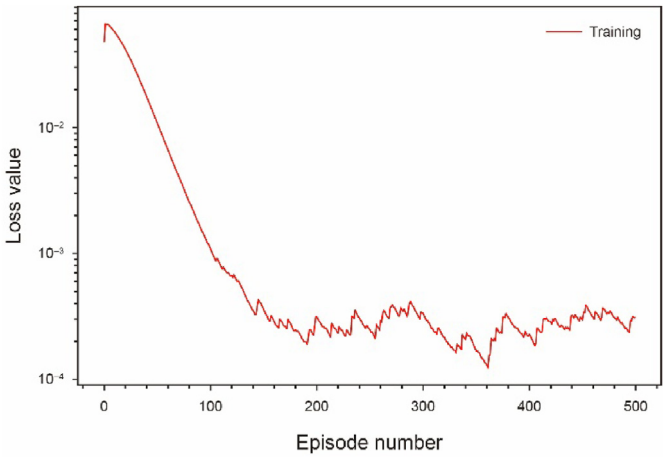

Fig. 7. The loss on the training set by our method.

Table 2
The test loss of all four methods.

Method	RMAML	Transfer learning	Mix training	Class-only
Test loss	0.0013	0.0096	0.015	0.011

enough to form the correct distribution shape, the network lacks enough gradient direction to update its parameters, which causes the prediction result deviates from the true value. Transfer learning works best compared with the above two methods because the network starts from more reasonable initial parameters that were learned from mixed learning through other classes of reservoirs. However, the initial parameters are still not good enough to shield interference from other classes. In our method, since the initial parameters are fully optimized without being affected by others, it is able to quickly approach better initial parameters of the network. After being tuned by query set data, the target sequence is accurately predicted.

4.2. Case2: EGG model

We try to simulate real application scenarios to test our method and the EGG model is a commonly used benchmark model. Specifically, it has 12 wells, including 4 production wells and 8 injection wells. The boundary of the entire model is elliptical and consists of 60*60*7 grids in total while the grid size is 8 m*8 m*4 m. The permeability field distribution shows typical fluvial facies deposition where channel and beach micro-facies can be clearly seen inside, as shown in Fig. 9. We test the accuracy of our method on this EGG model and complete the comparison with other methods, so as to obtain more realistic and reliable conclusions.

4.2.1. Given the condition and target

In order to better show the application results of our method on the EGG model. We assume that the boundary shape of the EGG model is known, and we plan to deploy 4 production wells, 8 water injection wells, and their corresponding injection-production regimes. The goal is to predict future production from the four producing wells without performing precise geological modeling.

4.2.2. Data collection and exhibition

Based on the boundary shape of the EGG model, we collected production data, including injection and recovery, and oil production, under various conditions through random permeability fields, random well locations, and random working regimes. The total number of sequences is 1000 and each contains 60 time steps in which each step's length is 30 days. They are shown in Fig. 10 after dimensionality reduction by PCA. It can be seen that the data points of these samples are relatively scattered, and there is no obvious data center. Moreover, these data points show the characteristics of clustering, which means that these data can be classified.

4.2.3. Data classification

After obtaining the data, we need to classify the data first, firstly, classify the training samples, and the classification method used is the K means method mentioned earlier. In order to make the distribution of each class of data more uniform, we finally divided the training data into 100 classes, and the classification results are shown in Fig. 11. In addition, because we ultimately need to predict the target data from specified reservoir permeability distribution and determined well pattern, we need to determine the class it belongs to base on its historical data. Specifically, we still use the K-means method used in the training data to predict the class of target data and add target data as a black point in Fig. 11. It is obvious that the target data is closest to the purple cluster, so the target data can be classified into the class to which the purple data belongs.

4.2.4. Model training and testing

Although our data is divided into many classes, when we divide the training set and the test set, we still mix different classes of samples together and then split them. It is worth noticing that we need to ensure that the samples in each batch belong to a different class. Specifically, we use 10% of the data as test data, 10% as validation data, and the remaining 80% as training data. During training, we use 3000 epochs to complete the training, and after every 50 training, a validation will be performed, and the optimal network parameters will be saved. The training loss curve is the red line shown in Fig. 12.

4.2.5. Application on target situation

To accomplish the final goal, we apply the trained network to the target data. In detail, we first select other samples (those purple sample points) that are in the same class as the target data and use the trained network to perform an inner loop based on these samples to update the network, so as to obtain the optimal parameters under the current class. Then directly import the input data of the target data to get the prediction result, as shown in the yellow line in Fig. 13. It can be clearly seen that the predicted yield is basically consistent with the actual yield.

4.2.6. Comparison with other methods

In order to reflect the advanced nature of our method, and to clarify the reasons for achieving some of these, we again compare the other three methods. From Table 3 and Fig. 13, we can see the prediction results of the four methods on the target data. The values of test loss on target data by these methods draw the same conclusion as training loss. Specifically, the method with the largest error is the mixed training method, followed by the class-only method. The closest to the results of our method is the transfer learning method. Here, we conclude that this happens mainly due to the following reasons. First, a large number of samples of different classes, especially those on the edge of the distribution, can cause misleading gradients to deviate from the optimal solution. Therefore, mixed training makes the training error the largest among these methods. Secondly, the loss of the class-only method is also relatively high, which is due to the unreasonable initial point of the network and too few samples. We can imagine that the network parameters converge from a random initial point. However, this random initial parameter may be very far from the global

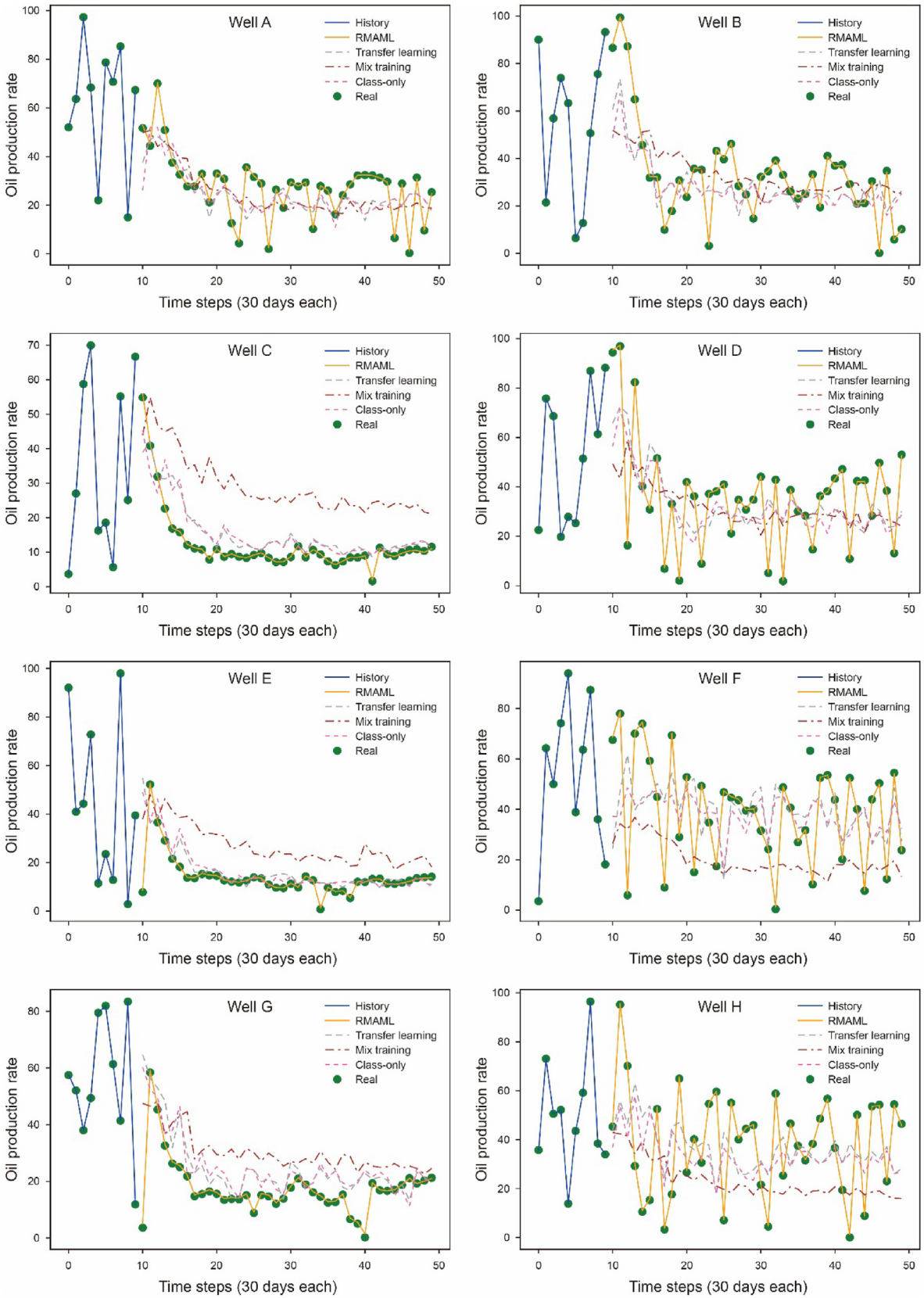

Fig. 8. The result predicted by many methods including our method.

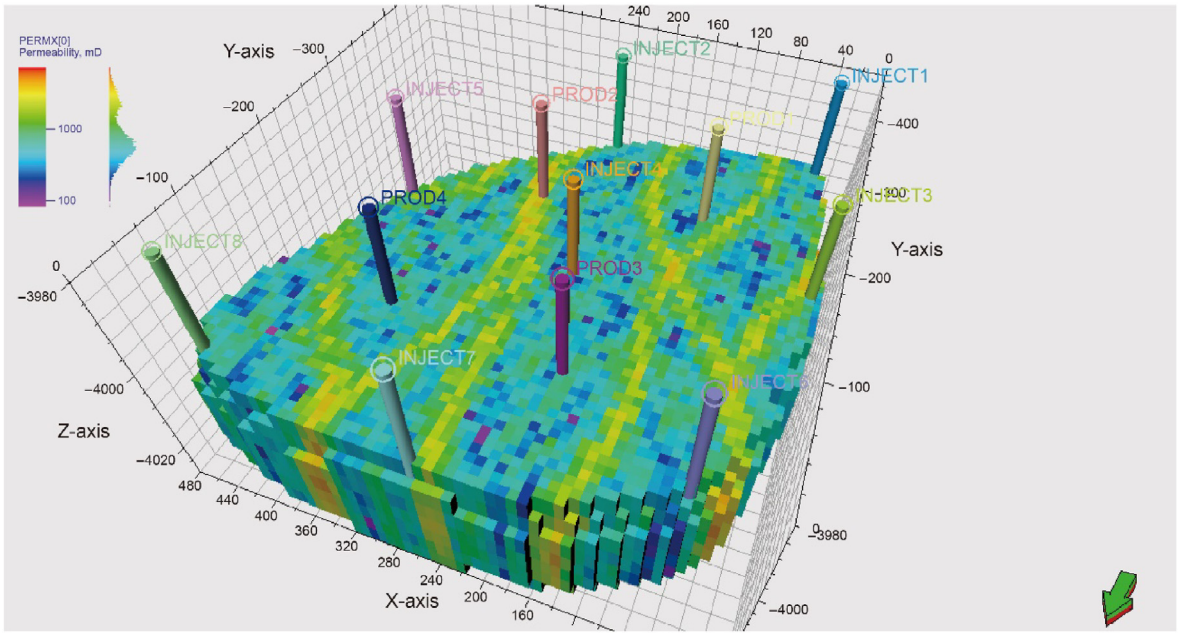

Fig. 9. The permeability field distribution of the EGG model.

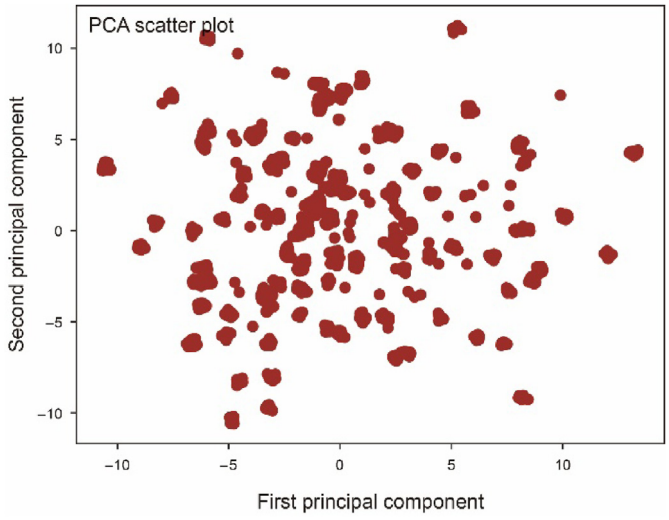

Fig. 10. The distribution of production data after PCA.

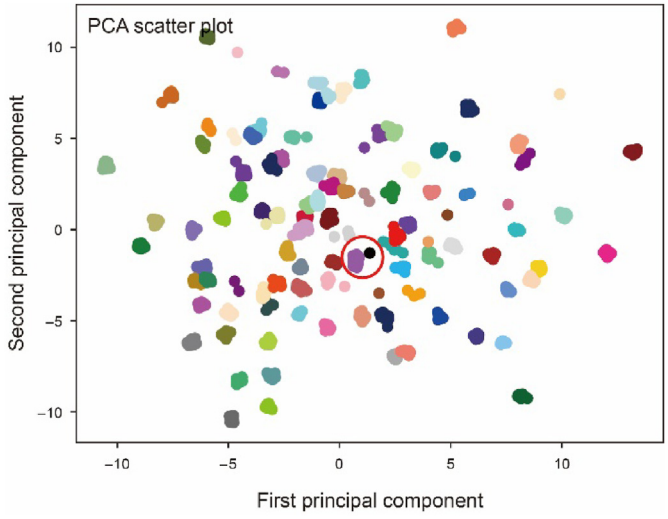

Fig. 11. Classes of data in the dataset including target data.

optimum, and the gradient provided by a small number of samples is very fixed. Therefore, this optimization process will fall into a local optimum region that is difficult to escape. Thirdly, the performance of the transfer learning method is better than the previous two methods, because it finds an initial state that is closer to the global optimal solution based on a large number of non-similar samples during pre-training, and then goes through the finetune process to quickly convergence. However, because this method uses the initial state obtained by mixing samples, it is inevitable to encounter mutual interference between the different classes mentioned above, which makes this initial point not good enough. Fourthly, as a very useful recursive algorithm, LSTM is widely used in the prediction of many kinds of time series data. However, the use of LSTM as a base learner is not a good decision because it is more sophisticated than our use of GRU as a base learner, which on the one hand increases computing time while on the other makes the overfitting phenomena worse. Finally, our method works best in the selection of initial points. As can be seen from the red line and orange line in Fig. 12, although the initial loss value of our method is larger than that of transfer during the training process, our method can converge faster and the loss after convergence is lower. Because our learn-to-learn method can be seen as a second-order network that is more flexible than the general one-order network, the mapping is more flexible and its solution is closer to the best result. Additionally, Recursive networks typically lose accuracy over time. The degree of oil production variance does, however, diminish as the development phase moves into the middle and late stages, which is a feature of reservoir production statistics that other common data do not have. This is brought on by the slowing down of the water content rise rate. Therefore, as development time increases, the difficulty of prediction gradually decreases, offsetting the intrinsic tendency of network prediction

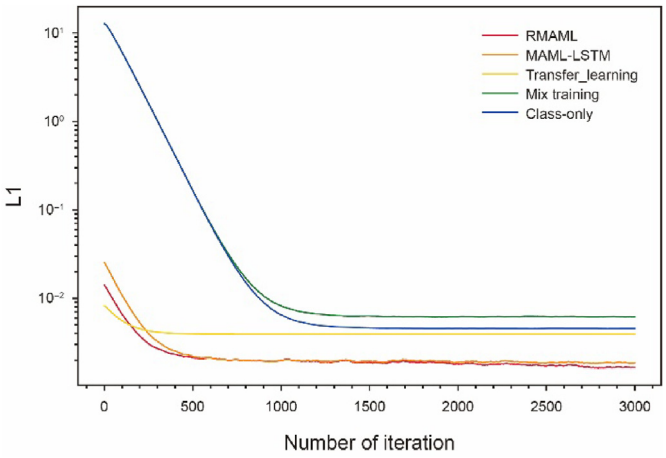

Fig. 12. The training loss curve from different methods.

accuracy dropping as time increases, giving the impression that the results of prediction are still quite accurate.

5. Conclusion

The RMAML method first obtains the corresponding parameters for each class by independently updating copied initial GRU network in each class. Then, the expectation of gradients is calculated based on the query data set in each class to improve the initial network parameters. Next, the optimal network parameters appear after training with a small number of samples in the same class as the target reservoir. Finally, future production indices of the target reservoir are accurately predicted with these network parameters. From its application in two cases, the following conclusions are summarized.

1. The distribution of a reservoir's production data varies depending on its permeability fields, well patterns, and production regimes, while the well pattern exerts the most influence.
2. Samples from various classes of reservoirs greatly distract the gradient direction of learning since the best network parameters for each class differ. Consequently, it has the largest prediction error at 0.0096 among these methods when the network is trained with huge amounts of samples from totally different reservoirs.
3. A small number of samples from the same class can not guarantee the network parameters converges to the global optimum. Due to the significant nonlinearity of the mapping itself, the gradient direction brought on by the lack of samples may make the optimization process stuck in a local suboptimal point.

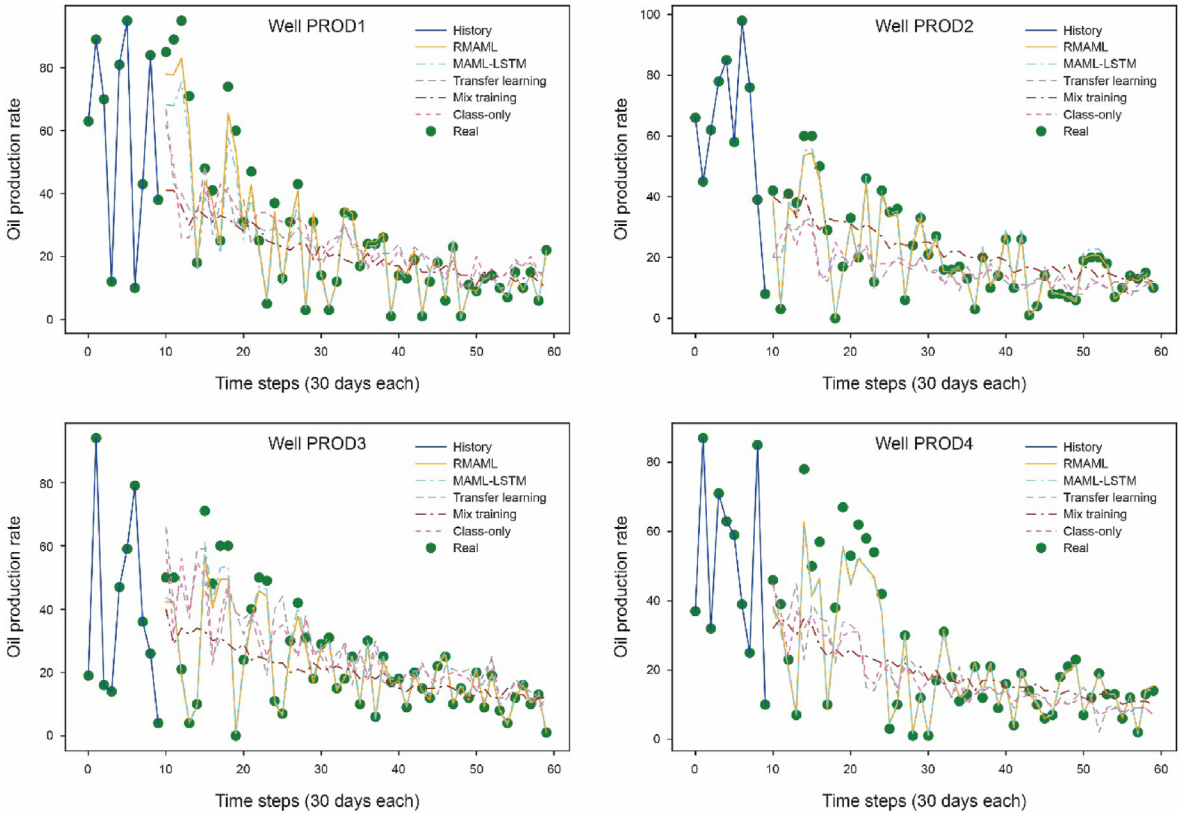

Fig. 13. The prediction of target data by different methods.

Table 3
The test loss on target data by different methods.

Method	RMAML	MAML-LSTM	Transfer learning	Mix training	Class_only
Target_loss	0.0016	0.0020	0.0075	0.0096	0.0088

4. The RMAML method can identify the variations and similarities among different classes, making it possible to efficiently learn the rules that are helpful for transferable prediction. The proposed RMAML method, in training, only requires a few samples from the same class to learn the special knowledge that is different from the common knowledge. As a result, its prediction error reduces to only 0.0016, which is nearly one-fifth of the error predicted by other methods.

6. Discussion

The method proposed in this article exhibits excellent generalization performance. However, changes in the dataset may lead to different computational results. The data samples used in this article were obtained from theoretical models of sandstone reservoirs with good permeability, without any on-site measurement errors, and with similar fluid properties. Therefore, the laws of these samples may be shared to a large extent, which makes the prediction results very good. However, when the data comes from reservoirs with significant differences in properties or with on-site measurement errors, although the method proposed in this article may yield better prediction results than conventional methods, it cannot guarantee perfect accuracy. Therefore, it is recommended that readers conduct data screening before using this method. Additionally, better data classification methods and dimension reduction techniques can also serve as helpful data preprocessing methods.

Declaration of competing interest

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

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant 52274057, 52074340 and 51874335, the Major Scientific and Technological Projects of CNPC under Grant ZD2019-183-008, the Major Scientific and Technological Projects of CNOOC under Grant CCL2022RCPS0397RSN, the Science and Technology Support Plan for Youth Innovation of University in Shandong Province under Grant 2019KJH002, 111 Project under Grant B08028.

References

Alimohammadi, H., Rahmanifard, H., Chen, N., 2020. Multivariate time series modelling approach for production forecasting in unconventional resources. In: SPE Annual Technical Conference and Exhibition. OnePetro. https://doi.org/10.2118/201571-MS.

Bakhty, N., Demin, A., Tupitsin, M., 2020. Assisted creation and usage of material balance models for production forecasting as a part of integrated field management. In: SPE Russian Petroleum Technology Conference. OnePetro. https://doi.org/10.2118/201958-MS.

Caena, F., Stringher, C., 2022. Towards a new conceptualization of Learning to Learn. J. Aula Abierta 49 (3), 199–216. https://doi.org/10.17811/rifie.49.3.2020.199-216.

Dehdari, V., Deutsch, C.V., 2020. Theory and implementation of an approximate physics simulator for heavy oil production forecasting. In: SPE Canada Heavy Oil Conference. OnePetro. https://doi.org/10.2118/199943-MS.

Dong, Y., Zhang, Y., Liu, F., et al., 2021. Reservoir production prediction model based on a stacked LSTM network and transfer learning. J. ACS omega 6 (50), 34700–34711. https://doi.org/10.1021/acsomega.1c05132.

Douglas, J., Furtado, F., Pereira, F., 1997. On the numerical simulation of waterflooding of heterogeneous petroleum reservoirs. J. Comput. Geosci. 1 (2), 155. https://doi.org/10.1023/A:1011565228179.

Du, E., Liu, Y., Cheng, Z., et al., 2022. Production forecasting with the interwell interference by integrating graph convolutional and long short-term memory neural network. J. SPE Reservoir Evaluat. Eng. 25 (2), 197–213. https://doi.org/10.2118/208596-PA.

Gao, N., Ziesche, H., Vien, N.A., Volpp, M., Neumann, G., 2022. What matters for meta-learning vision regression tasks?. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14776–14786. https://doi.org/10.1109/CVPR52688.2022.01436.

Gupta, I., Samandarli, O., Burks, A., et al., 2021. Autoregressive and machine learning driven production forecasting-Midland Basin case study. In: SPE/AAPG/SEG Unconventional Resources Technology Conference. OnePetro. https://doi.org/10.15530/urtec-2021-5184.

Hakim Elahi, S., 2019. A novel workflow for oil production forecasting using ensemble-based decline curve analysis. In: SPE Annual Technical Conference and Exhibition. OnePetro. https://doi.org/10.2118/195916-MS.

Iwata, T., Kumagai, A., 2020. Meta-learning from tasks with heterogeneous attribute spaces. J. Advances in Neural Information Processing Systems 33, 6053–6063.

Jolliffe, I.T., Cadima, J.A.M., 2016. Principal component analysis: a review and recent developments. J. Phys. Sci. E. 374 (2065), 20150202. https://doi.org/10.1098/rsta.2015.0202.

Kocoglu, Y., Gorell, S., McElroy, P., 2021. Application of Bayesian optimized deep Bi-LSTM neural networks for production forecasting of gas wells in unconventional shale gas reservoirs. In: Unconventional Resources Technology Conference, 26–28 July 2021, pp. 2176–2196. https://doi.org/10.15530/urtec-2021-5418.

Lemke, C., Gabrys, B.J.N., 2010. Meta-learning for time series forecasting and forecast combination. J. Neurocomput. 73 (10-12), 2006–2016. https://doi.org/10.1016/j.neucom.2009.09.020.

Li, Y., Guan, C., Guan, C., et al., 2019. An optimized neural network prediction model for gas assisted gravity drainage recovery based on dimensional analysis. Petroleum Science Bulletin 03, 288–299.

Li, Y., Han, Y., 2017. Decline curve analysis for production forecasting based on machine learning. In: SPE Symposium: Production Enhancement and Cost Optimisation. OnePetro. https://doi.org/10.2118/189205-MS.

Li, M., Tang, Y., Ma, W., 2022. Few-shot traffic prediction with graph networks using locale as relational inductive biases. J. IEEE Transact. Intelligent Transport. Syst. https://doi.org/10.1109/TITS.2022.3219618.

Lu, L., Xiong, S., 2022. Few-shot driver identification via meta-learning. J. Expert Syst. Appl. 203, 117299. https://doi.org/10.1016/j.eswa.2022.117299.

Ma, Y., Zhao, S., Wang, W., et al., 2022. Multimodality in meta-learning: a comprehensive survey. J. Knowledge-Based Systems, 108976. https://doi.org/10.1016/j.knosys.2022.108976.

Makinde, I., Lee, W.J., 2016. Production forecasting in shale volatile oil reservoirs using reservoir simulation, empirical and analytical methods. In: SPE/AAPG/SEG Unconventional Resources Technology Conference. OnePetro. https://doi.org/10.15530/URTEC-2016-2429922.

Masini, S.R., Goswami, S., Kumar, A., Chennakrishnan, B., Baghele, A., 2020. Artificial intelligence assisted production forecasting and well surveillance. In: Offshore Technology Conference Asia. OnePetro. https://doi.org/10.4043/30332-MS.

Mohd, R., Cornelio, J., Cho, Y., et al., 2022. Transfer learning with recurrent neural networks for long-term production forecasting in unconventional reservoirs. J. SPE 27 (4), 2425–2442. https://doi.org/10.2118/209594-PA.

Nolen, J., 1973. Numerical simulation of compositional phenomena in petroleum reservoirs. In: SPE Symposium on Numerical Simulation of Reservoir Performance. OnePetro. https://doi.org/10.2118/4274-MS.

Odi, U., Ayeni, K., Alsulaiman, N., et al., 2021. Applied transfer learning for production forecasting in shale reservoirs. In: SPE Middle East Oil & Gas Show and Conference. OnePetro. https://doi.org/10.2118/204784-MS.

Oreshkin, B.N., Dudek, G., Pełka, P., et al., 2021. N-BEATS neural network for mid-term electricity load forecasting. J. Appl. Energy 293, 116918. https://doi.org/10.1016/j.apenergy.2021.116918.

Razak, S., Cornelio, J., Cho, Y., et al., 2021. A physics-guided deep learning predictive model for robust production forecasting and diagnostics in unconventional wells. In: Unconventional Resources Technology Conference, 26–28 July 2021, pp. 1843–1850. https://doi.org/10.15530/urtec-2021-5059.

Simon, C., Koniusz, P., Nock, R., et al., 2020. Adaptive subspaces for few-shot learning. In: Conference on Computer Vision and Pattern Recognition (CVPR). IEEE, pp. 4135–4144. https://doi.org/10.1109/CVPR42600.2020.00419.

Song, S., Hong, B., Shi, B., et al., 2017. Research into calculation of natural gas well production based on an artificial neural network. Petroleum Science Bulletin 2 (3), 413–421.

Temizel, C., Canbaz, C., Saracoglu, O., et al., 2020. Production forecasting in shale reservoirs through conventional DCA and machine/deep learning methods. In: Unconventional Resources Technology Conference, 20–22 July 2020, pp. 4843–4894. https://doi.org/10.15530/urtec-2020-2878.

Wang, Q., Yuan, C., Liu, Y., 2019. Learning deep conditional neural network for image segmentation. J. IEEE Transact. Multimedia 21 (7), 1839–1852. https://doi.org/10.1109/TMM.2018.2890360.

Yang, N., Zhang, B., Ding, G., et al., 2021. Specific emitter identification with limited samples: a model-agnostic meta-learning approach. J. IEEE Commun. Lett. 26 (2), 345–349. https://doi.org/10.1109/LCOMM.2021.3110775.

Zhong, Z., Wu, X., 2017. Pressure and saturation relationship in CBM reservoir and the semi-analytical production forecast model for the horizontal well. Petroleum Science Bulletin 2 (2), 251–257.