


## Article

# A Physics-Informed Neural Network Approach for Surrogating a Numerical Simulation of Fractured Horizontal Well Production Prediction

Taiyu Jin <sup>1</sup>, Yang Xia <sup>1,\*</sup> and Haolin Jiang <sup>2,\*</sup>

<sup>1</sup> National Key Laboratory of Petroleum Resources and Engineering, China University of Petroleum (Beijing), Beijing 102249, China

<sup>2</sup> School of Integrated Circuit Science and Engineering, University of Electronic Science and Technology of China (UESTC), Chengdu 611731, China

\* Correspondence: xiayang@vip.126.com (Y.X.); orrinj@gmail.com (H.J.)

**Abstract:** With the popularity of deep learning (DL), more and more studies are focusing on replacing time-consuming numerical simulations with efficient surrogate models to predict the production of multi-stage fractured horizontal wells. Previous studies on constructing surrogate models for the prediction of the production of fractured horizontal wells often relied on directly applying existing deep learning architectures without incorporating physical constraints into the model. When dealing with the large number of variables necessary for characterizing the properties of fractures, the input variables of proxy models are often oversimplified; meanwhile, lots of physical information is lost. Consequently, predictions are sometimes physically inconsistent with the underlying principles of the domain. In this study, by modifying the traditional Seq2Seq (LSTM–LSTM) deep learning architecture, a physics-informed encoder–decoder (PIED) architecture was developed to surrogate the numerical simulation codes for predicting the production of horizontal wells with unequal-length intersecting hydraulic fractures on a 2D plane. The encoder is a LSTM network, and the decoder consists of LSTM and fully connected layers. The attention algorithm is also applied in the Seq2Seq architecture. The PIED model’s encoder is capable of extracting the physical information related to fractures. And the attention module effectively passes on the most relevant physical information related to production to the decoder during the training process. By modifying Seq2Seq architecture, the decoder of the PIED incorporates the intermediate input, which is the constant production time, along with the extracted physical information to predict production values. The PIED model excels in extracting sufficient physical information from high-dimensional inputs while ensuring the integrity of the production time information. By considering the physical constraints, the model predicts production values with improved accuracy and generalization capabilities. In addition, a multi-layer perceptron (MLP) which is broadly used as a proxy model; a regular Seq2Seq model (LSTM–Attention–LSTM); and the PIED were compared via a case study, and their MAE values were shown to be 241.76, 184.07, 168.81, respectively. Therefore, the proposed model has higher accuracy and better generalization ability. In the case study, a comparative experiment was conducted by comparing LSTM–MLP (with an MAE of 221.50) and LSTM–LSTM to demonstrate that using LSTM as the decoder structure is better for predicting production series. Moreover, in the task of predicting production sequences, LSTM outperforms MLP. The Seq2Seq architecture demonstrated excellent performance in this problem, and it achieved a 48.4% reduction in MSE compared to MLP. Meanwhile, the time cost for build datasets was considered, and the proposed model was found to be capable of training in a small dataset (e.g., in the case study, 3 days were used to generate 450 samples for training.); thus, the proposed model has a certain degree of practicality.

**Keywords:** fractured well; production; proxy model; physics-informed neural network; deep learning; machine learning



**Citation:** Jin, T.; Xia, Y.; Jiang, H. A Physics-Informed Neural Network Approach for Surrogating a Numerical Simulation of Fractured Horizontal Well Production Prediction. *Energies* **2023**, *16*, 7948. <https://doi.org/10.3390/en16247948>

Academic Editor: Reza Rezaee

Received: 23 June 2023

Revised: 19 July 2023

Accepted: 2 August 2023

Published: 7 December 2023



**Copyright:** © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

### 1.1. Traditional Numerical Simulation of Fractured Horizontal Well Production

Hydraulic fracturing is a treatment widely applied to enhance oil and gas recovery, especially in low-permeability reservoirs. The artificial fractures created during hydraulic fracturing substantially improve the seepage resistance of fluid flow in a reservoir. The success of hydraulic fracturing in increasing production depends largely on the geometry of the hydraulic fractures, so before fracturing is implemented, numerical simulation methods are usually required to predict the geometry of the hydraulic fractures and the fracture conductivity, as well as other parameters, and then calculate important indices such as well production to evaluate or optimize the fracturing schemes.

Currently, the main method for fractured horizontal well production prediction is the numerical simulation method. Numerical simulation methods usually include the finite difference method (FDM), the finite element method (FEM), the boundary element method (BEM), and the discrete element method (DEM). Due to the multiple parameters involved (e.g., fracture permeability, fracture inclination, fracture length, etc.), it is time-consuming to perform high-accuracy production prediction using numerical simulation methods. To optimize hydraulic fracturing parameters and thus maximize production, traditional trial-and-error methods tend to be very time-consuming and expensive. Therefore, there is a need to develop some proxy models for numerical simulation methods for hydraulic fracturing production prediction. Proxy models are approximations of numerical simulation models, usually based on machine learning algorithms, which map the inputs and outputs of the numerical simulation model. Proxy models for fractured well production prediction can be efficiently performed to significantly reduce computational costs and speed up the optimization process in large-scale parameter space search and optimization.

### 1.2. Review of Research

In recent years, with the popularity of machine learning, many studies have been performed on data-driven machine learning models to surrogate physically driven numerical simulations for fractured well production. These studies simplify the input data or output data of numerical simulators and then use them to build datasets and train machine learning models. Gaussian process regression (GPR), convolutional neural networks (CNNs), and support vector machines (SVMs) have been used as proxy models for numerical simulation models of fractured horizontal wells with parallel, equal-length hydraulic fractures. These proxy models have been used to predict gas production and to obtain optimal fracture half-lengths and horizontal well lengths (Wang et al., 2021) [1]. Multi-layer perceptron (MLP) can be used to construct surrogate models to predict the production of gas wells with parallel, equal-length hydraulic fractures on a two-dimensional plane (Wang et al., 2021) [2]. The tree-based ensemble method was used as a proxy model of a numerical simulator of 2D discrete fracture networks. This proxy model can predict 5-year oil cumulative production (Xue et al., 2019) [3]. The regular neural network (NN), random forests (RFs), adaptive boosting (AdaBoost), and support vector machines (SVMs) were used to build several proxy models of numerical simulation codes which can predict the production and the net present value (NPV) of a horizontal well with parallel and unequal-length hydraulic fractures (Li et al., 2022) [4]. Transformer, a Seq2Seq DL model, was used to predict the time series of production data based on the integration of geology and engineering time series production data (Wang et al., 2023) [5].

However, the input variables of the proxy models mentioned in these studies are often oversimplified. For example, among the previously mentioned proxy models, CNN, GPR, and SVM (Wang et al., 2021) [1] require that the length of the hydraulically fractured fractures in the dataset take only a fixed number of four values; the training set of the MLP network (Wang et al., 2021) [2] contains only four variables, and these four features can only take a fixed number of four values; the tree-based ensemble model (Xue et al., 2019) [3] requires that the dataset can only have these four variables, which can only take a fixed number of three values; and the input data of the transformer (Wang et al. [5]) need to be

preprocessed by PCA (principal component analysis) to decrease its dimension, but the variables generated by PCA often pose challenges in terms of interpretation. Although reducing the dimensions of the input data or simplifying the input data can reduce the complexity of machine learning models and make training easier, this also decreases the performance of the proxy models.

Incorporating prior physical knowledge into neural networks empowers the proxy model to be easily trained in the presence of numerous input variables, endowing it with enhanced accuracy and generalization capabilities [6–8]. Physics-informed neural networks have the advantages of increased learning efficiency [9], improved generalization [10], enhanced interpretation, and improved robustness in solving physical problems [11], and this is supported by relevant studies.

Therefore, the machine learning architecture of the proxy models for fractured well production prediction needs to be adapted to the specific engineering problem by introducing physical information, rather than simply applying machine learning algorithms. For example, a physics-constrained Bi-GRU-DHNN method was used to predict the production of hydraulic fractured wells (Li et al., 2022) [12]. The model consists of two parts. The first part consists of MLP, CNN, and Bi-GRU. Different network structures in the first part can extract different kinds of features (i.e., ecological properties, logging curves, well information, fracturing parameters, and field operations) and generate a preliminary prediction value as well as two feature vectors of physical information. In the second part, a MLP can establish a nonlinear mapping between the physical constraints and the production target. The model can input very high-dimension features which include six numerical variables and ten sequence variables. A physics-informed MLP (Li et al., 2021) [4] was developed as a proxy model of numerical simulation codes which could predict the production of a horizontal well with parallel- and unequal-length hydraulic fractures. The proxy model was introduced to prior physical knowledge by pre-training it with a large number of variables of all the fractures and then training the model with the small number of input variables that remained. The authors of this study demonstrate in the paper that the MLP integrated with prior physical knowledge has a stronger generalization ability and higher accuracy than a normal MLP. Guevara et al. [13] used generalized additive models (GAMs) to predict the cumulative production of fractured wells. By using GAMs, they could modify the smooth functions of each input variable to align the relationship between these input variables and the output with prior knowledge, thereby improving the accuracy of the results. Qu et al. [14] proposed a physics-informed MLP which could combine information regarding the static parameters of a fractured well with the input variables to predict fracture parameters. Yang et al. [15] developed a hybrid NN called GRU–MLP. They used MLP to combine the static parameters of hydraulic fractures with the production predicted by GRU, so that the production predicted by MLP was physically constrained and had a lower bias. Wang et al. [16] proposed a theoretical guide to CNN to surrogate the numerical simulation of two-phase flow in fields by using the coupled and discretized equations as loss functions in the training process. Cornelio et al. [17] used an auto encoder (AE) to extract the physical information of the hydraulic fractures and the reservoir properties. Then, they combined the physical information outputted from AE with the production predicted by a numerical simulator into an MLP to predict the production of fractured wells. Razak et al. [18] leveraged neural networks to rectify the errors in production calculated by parsing formulas, thus constraining the model's predictive values with physical knowledge. These physics-informed models can exhibit higher accuracy as well as learning significantly more features than ML models without physical constraints.

### 1.3. A Novel Proxy Model

In these previous studies, the numerical simulations surrogated by neural networks did not consider the influence of the geometry and permeability of each hydraulic fracture on production. Therefore, their proxy models cannot be used for the optimization of the ge-

ometric parameters of hydraulic fractures. In this study, we developed a physics-informed neural network proxy model for a numerical simulator that can calculate production based on the geometric parameters and permeability of each hydraulic fracture. The proxy model, called a physics-informed encoder–decoder (PIED), was developed for the numerical simulation of the production prediction of horizontal wells with non-equal-length intersecting hydraulic fractures. The proxy model has a modified Seq2Seq architecture and consists of an encoder and a decoder.

To develop this proxy model, the Seq2Seq structure was modified in this study so that the decoder could have the intermediate input of a static time series.

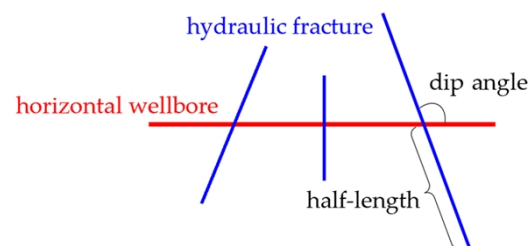
The proxy model utilizes the encoder structure and the attention module to integrate physical constraints into the decoder structure for predicting production, and it is robust enough to handle fracture intersections, so that it can predict production accurately in complex physical scenarios.

## 2. Data Preparation from Numerical Simulations

Before performing the machine learning model, the input and output data should be carefully prepared, and a large dataset is required to mine the relationship between the inputs and the outputs. In this section, the characterization of a fracture, a coupled reservoir flow model, and the numerical simulation method are proposed to prepare the required inputs and outputs.

### 2.1. Fracture Characterization

In the discrete fracture model, the fracture is modelled as a zero-thickness plate or a slit [19]. The schematic diagram of the fractured horizontal well is shown in Figure 1. The morphology of each hydraulic fracture is different, varying in two geometric parameters: the half-length and dip angle, shown in Figure 1. During the fracturing process, the amount of sand injected into each fracture is different, so the permeability of each hydraulic fracture is different. Therefore, three parameters were selected to characterize the hydraulic fracture and serve the inputs into the network. The dip angle, half-length and permeability of the fracture were evaluated by a uniform variation of  $(60\text{--}120)^\circ$ ,  $(5\text{--}75)$  m,  $(1\text{--}1000)$  D.



**Figure 1.** A schematic diagram of the fractured horizontal well.

### 2.2. Reservoir Flow Model

The basic governing equations for fluid flow in a fractured reservoir are [20]:

$$\frac{\partial(\rho\phi_m)}{\partial t} + \nabla \cdot (\rho v_m) = 0 \quad (1)$$

$$\frac{\partial(\rho\phi_f)}{\partial t} + \nabla \cdot (\rho v_f) = 0 \quad (2)$$

where  $\rho$ ,  $\phi$ , and  $v$  are the pore fluid density, porosity, and Darcy velocity, respectively. The subscripts  $m$  and  $f$  represent the physical parameters of the matrix and fracture, respectively. The Darcy velocity is given by:

$$v_{m,f} = -\frac{\kappa_{m,f}}{\mu} \nabla p_{m,f} \quad (3)$$

where  $\kappa$  and  $\mu$  are the reservoir permeability and fluid viscosity. The left sides of Equations (1) and (2) can be rewritten as [21]

$$\frac{\partial(\rho\phi_{m,f})}{\partial t} = c_{tm,f}\rho_0\frac{\partial p_{m,f}}{\partial t} \quad (4)$$

with the total compressibility  $c_{tm,f}$  considered to be a lumped, constant parameter which contains both the compressibility of the solid and the fluid. After substituting Equations (3) and (4) into Equations (1) and (2), the governing equations for coupled flow in a fractured reservoir can be obtained:

$$\frac{\partial p_{m,f}}{\partial t} = \gamma_{m,f}\nabla^2 p_{m,f} \quad (5)$$

where  $\gamma_{m,f} = \kappa_{m,f}/\mu c_{tm,f}$ . The total compressibility can be related to the rock modulus and fluid modulus:

$$c_{tm,f} = \frac{\phi}{K_{ff}} + \frac{1}{K_{m,f}} \quad (6)$$

where  $K_{ff}$  is the fluid modulus and  $K_{m,f}$  is the modulus of the matrix or fracture. When the reservoir is considered fully saturated, the total mass of the drained fluid, i.e., the cumulative production, can be expressed by:

$$M_q = \int (\rho_0\phi_{m0} - \rho\phi_m)d\Omega \quad (7)$$

where the subscript 0 represents the initial value. Using the state equations, the cumulative mass production can be calculated by:

$$M_q = \rho_0 c_{tm} \int (p_{m0} - p_m)d\Omega \quad (8)$$

### 2.3. Weak Formulations

The Gauss divergence theorem is used for discrete formulations [22]:

$$\int_{\Omega} \text{div} F d\Omega = \int_{\Gamma} F \cdot n_{\Gamma} d\Gamma - \sum_{i=1}^N \int_{f_{r_i}} F^{\pm} \cdot n_{f_{r_i}} d\Gamma \quad (9)$$

where  $F$  represents a discontinuous function, and  $n_{\Gamma}$  and  $n_{f_r}$  are the outward normal directions of the external boundary and fracture surface, respectively. The jump generated by the discontinuous function at the fracture surface is denoted by  $F^{\pm}$ ,  $F^{\pm} = F^+ - F^-$ .  $F^+$  and  $F^-$  represent the function values on the positive and negative sides of the fracture, respectively. To derive the weak form of the matrix flow equation, we multiply the strong form by a test function  $\delta p$  and construct an equivalent integral form. Then, we utilize the Gauss divergence theorem to derive the weak form.

$$\int_{\Omega} \delta p \frac{\partial p_m}{\partial t} d\Omega + \sum_{i=1}^N \int_{f_{r_i}} \gamma_m \delta p (\nabla p_m \cdot n_{f_{r_i}})^{\pm} d\Gamma + \int_{\Omega} \gamma_m \nabla(\delta p) \cdot \nabla p_m d\Omega + \int_{\Gamma_q} \frac{\delta p \bar{q}}{c_{tm}} d\Gamma = 0 \quad (10)$$

$d\Omega$  represents the reservoir domain, while  $\Gamma_q$  is a constant flow rate boundary. On  $\Gamma_q$ , the flow rate  $q = \bar{q}$ . The third term on the left side of the equation is the discontinuous term. By combining with Darcy's law, we can obtain the following:

$$\int_{f_{r_i}} \gamma_m \delta p (\nabla p_m \cdot n_{f_{r_i}})^{\pm} d\Gamma = \int_{f_{r_i}} \frac{\delta p}{c_{tm}} \left( \frac{\kappa_m}{\mu} \nabla p_m \right)^{\pm} \cdot n_{f_{r_i}} d\Gamma = - \int_{f_{r_i}} \frac{1}{c_{tm}} \delta p (n_{f_{r_i}} \cdot q_f) d\Gamma \quad (11)$$

where  $q_f$  is used to characterize the fluid exchange between fractures and the matrix. The exchange term  $q_f$  can be obtained by solving the flow governing equation within the fractures. To derive the weak form of the fracture flow equation, we multiply the strong

form by a test function  $\delta p$  and construct an equivalent integral form. Then, we utilize the Gauss divergence theorem to derive the weak form.

$$\int_{\Omega_{fi}} \delta p \frac{\partial p_f}{\partial t} d\Omega + \int_{\Omega_{fi}} \gamma_f \nabla(\delta p) \cdot \nabla p_f d\Omega - \int_{f_{\Gamma_i}} \gamma_f \delta p \nabla p_f \cdot n_{f_{\Gamma_i}} d\Gamma = 0 \quad (12)$$

Considering that the fracture's width is significantly smaller than its length, the pressure variation in the width direction can be disregarded. By combining this with Darcy's law, the expression for the exchange term  $q_f$  can be derived as follows:

$$\int_{f_{\Gamma_i}} \delta p (n_{f_{\Gamma_i}} \cdot q_f) d\Gamma = -c_{tf} \left( \int_{f_{\Gamma_i}} d_{fi} \delta p \frac{\partial p_f}{\partial t} d\Gamma + \int_{f_{\Gamma_i}} d_{fi} \gamma_f \frac{\partial \delta p}{\partial x} \frac{\partial p_f}{\partial x} d\Gamma \right) \quad (13)$$

By substituting the exchange term between the matrix and fractures (Equation (13)) into the weak form of matrix flow (Equation (10)), we can obtain the weak form of the matrix fracture coupled flow equation.

$$\int_{\Omega} \frac{\delta p}{\gamma_m} \frac{\partial p_m}{\partial t} d\Omega + \int_{\Omega} \nabla(\delta p) \nabla p_m d\Omega + \sum_{i=1}^n \int_{f_{\Gamma_i}} \left( \frac{d_{fi}^{k_i}}{\kappa_m} \frac{\partial(\delta p)}{\partial x^i} \frac{\partial p_m}{\partial x^i} + \frac{d_{fi}^{c_{tf}}}{\gamma_m c_{tm}} \delta p \frac{\partial p_m}{\partial t} \right) d\Gamma + \int_{\Gamma_q} \frac{\delta p \bar{q}}{\gamma_m c_{tm}} d\Gamma = 0 \quad (14)$$

In Equation (14), a line integral along the fracture segment is performed (the third item on the left of the equal sign); this action is related to the fracture length and fracture dip.

#### 2.4. XFEM Solver

The discrete fracture model is commonly solved using the finite element method (FEM), finite volume method (FVM), boundary element method (BEM) and extended finite element method (XFEM). In the FEM and FVM, unstructured grids are used to discretize the reservoir domain, and nodal points are set on the fracture segment to maintain the flow consistency between the porous matrix and the fractures [23,24]. These methods can discretize a geometrically complex reservoir when using mesh points optimally and exactly represent the flow in the matrix and fractures; however, the fracture segment is restricted to the inter-element boundaries and the problem of mesh dependency limits their application [25,26]. The BEM can reduce the dimension of the problem, but the flow distribution around the fracture junction always results in computational difficulties [27]. To eliminate the requirement for the mesh topology to conform with that of the fracture network, an XFEM was developed that relies on adding enrichment functions to enhance the conventional finite element interpolation [28]. In the XFEM, a structured mesh can be used for explicit simulation on complex fracture networks; thus, the XFEM can overcome the constraint of mesh conformity between the reservoir and the fractures [26,29]. During the data preparation from numerical simulations, the fracture geometry changes in each simulation example, it becomes convenient to run multiple simulations with different fracture configurations without meshing the domain each time. Therefore, XFEM was used in this study for the numerical simulation.

In our XFEM solver, a structured grid is employed to discretize the matrix. The fractures in the computational domain are cut off by the element edges and the fracture segments can be classified as belonging to two groups: (I) fracture segments that completely cut an element and (II) fracture segments that partly cut an element, as shown in Figure 2.

Based on the partition of unit method, additional terms are added to the standard finite element approximation of the pressure:

$$p = \sum_{i \in N_s} N_i p_i + \Psi(x) = \sum_{i \in N_s} N_i p_i + \sum_{j \in N_{enr}} \varphi_j(x) \tilde{p}_j \quad (15)$$

where  $N_i$  is the standard FEM shape function;  $p_i$  represents the standard pressure degrees of freedom for nodes  $N_s$ .  $\tilde{p}_j$  is the added enriched degrees of freedom for nodes  $N_{enr}$ .  $\varphi_j(x)$  is the superimposed enrichment function:

$$\varphi_j(x) = N_j(x) [\Phi^n(x) - \Phi^n(x_j)], n = I, II \tag{16}$$

where the corresponding enrichment functions for the two types of fracture segments are:

$$\Phi^I(x) = \sum_i |f(x)| N_i(x) - \left| \sum_i f(x) N_i(x) \right| \tag{17}$$

$$\Phi^{II}(x) = \sqrt{r} \cos \frac{\theta}{2} \tag{18}$$

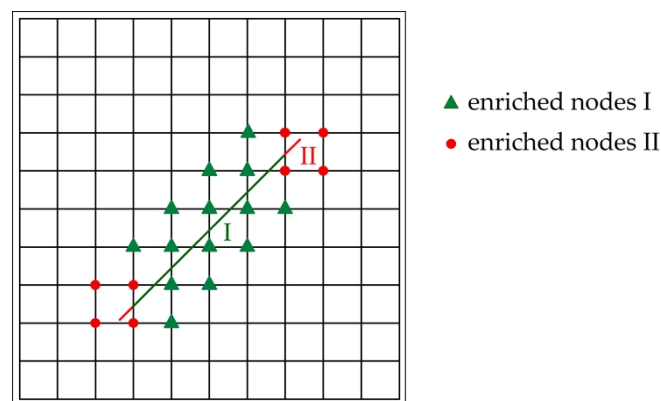
where  $f$  is the level set function, and  $r$  and  $\theta$  are the local polar coordinates near the fracture tip. By substituting Equation (15) into the weak form (14), we obtain the discretized governing equation:

$$\begin{bmatrix} K^{pp} & K^{p\tilde{p}} \\ K^{\tilde{p}p} & K^{\tilde{p}\tilde{p}} \end{bmatrix} \begin{pmatrix} p \\ \tilde{p} \end{pmatrix} + \begin{bmatrix} M^{pp} & M^{p\tilde{p}} \\ M^{\tilde{p}p} & M^{\tilde{p}\tilde{p}} \end{bmatrix} \begin{pmatrix} \dot{p} \\ \dot{\tilde{p}} \end{pmatrix} = \begin{pmatrix} F \\ \tilde{F} \end{pmatrix} \tag{19}$$

where

$$\begin{aligned} K_{IJ}^{mn} &= \int_{\Omega^e} B_I^m (B_J^n)^T d\Omega + \sum_{i=1}^N \int_{\Gamma_i} \frac{d_f^i k_f^i}{\kappa_m} B_I^{mi} (B_J^{ni})^T d\Gamma \quad (m, n = p, \tilde{p}) \\ \begin{cases} M_{IJ}^{pp} &= \int_{\Omega^e} \frac{1}{\gamma_m} N_I N_J^T d\Omega + \sum_{i=1}^N \int_{\Gamma_i} \frac{d_f^i c_{tf}^i}{\gamma_m c_{tm}^i} N_I N_J^T d\Gamma \\ M_{IJ}^{\tilde{p}\tilde{p}} &= (M_{IJ}^{\tilde{p}\tilde{p}})^T = \int_{\Omega^e} \frac{1}{\gamma_m} \varphi_I \varphi_J^T d\Omega + \sum_{i=1}^N \int_{\Gamma_i} \frac{d_f^i c_{tf}^i}{\gamma_m c_{tm}^i} N_I \varphi_J^T d\Gamma \\ M_{IJ}^{\tilde{p}p} &= \int_{\Omega^e} \frac{1}{\gamma_m} \varphi_I \varphi_J^T d\Omega + \sum_{i=1}^N \int_{\Gamma_i} \frac{d_f^i c_{tf}^i}{\gamma_m c_{tm}^i} \varphi_I \varphi_J^T d\Gamma \end{cases} \\ \begin{cases} F_I &= \int_{\Gamma_q} N_I \bar{q} d\Gamma \\ \tilde{F}_I &= \int_{\Gamma_q} \varphi_I \bar{q} d\Gamma \end{cases} \end{aligned}$$

where  $B = \nabla N$ .



**Figure 2.** Two types of fracture segments and the enriched nodes (the green segment completely crosses the element, and the red segment partly crosses the element).

### 2.5. Inputs and Outputs

Ten hydraulic fractures are simulated. During the simulations, the fracture geometry and permeability change while the matrix and fluid properties are fixed. For all the simulation cases, the fixed parameters are listed in Table 1, and the inputs and outputs of a simulation example are listed in Table 2. Presentations of mesh discretization, the

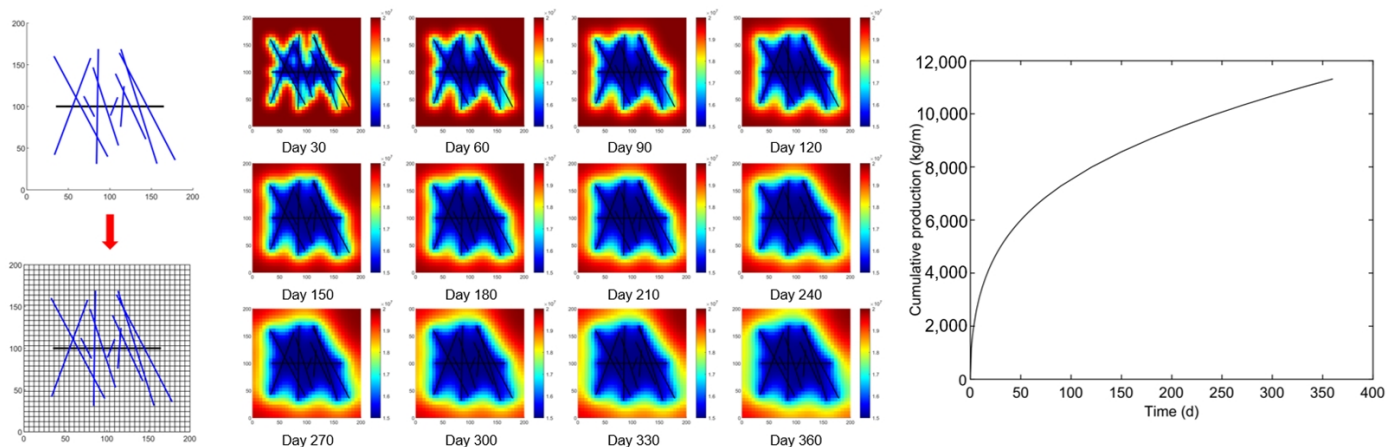
calculated pressure distributions at different times, and the cumulative production curve are shown in Figure 3.

**Table 1.** Fixed parameters across all simulations.

Parameters	Value	Unit
Matrix permeability	$5 \times 10^{-18}$	$\text{m}^2$
Fluid modulus	$3 \times 10^9$	Pa
Matrix modulus	$20 \times 10^9$	Pa
Fracture modulus	$1.05 \times 10^9$	Pa
Fluid viscosity	$3 \times 10^{-3}$	Pa·s
Initial fluid density	1000	$\text{kg}/\text{m}^3$
Porosity	0.15	/
Wellbore pressure	15	MPa
Reservoir pressure	20	MPa
Reservoir thickness	1	m

**Table 2.** Inputs and outputs of a simulation example.

Inputs			Outputs			
Fracture Half-Length (m)	Fracture Dip	Fracture Permeability ( $\text{m}^2$ )	Cumulative Production (kg)			
62.03	69.45°	$6.65 \times 10^{-10}$	Day 1	1164.32	Day 120	7984.58
68.41	118.23°	$1.25 \times 10^{-10}$	Day 3	1905.52	Day 150	8567.65
13.89	117.43°	$3.22 \times 10^{-12}$	Day 6	2591.89	Day 180	9074.56
68.94	89.12°	$1.37 \times 10^{-12}$	Day 9	3086.58	Day 210	9527.55
49.27	108.01°	$1.78 \times 10^{-9}$	Day 15	3814.71	Day 240	9939.82
11.83	68.51°	$7.59 \times 10^{-10}$	Day 21	4359.01	Day 270	10,319.74
24.49	85.30°	$6.56 \times 10^{-10}$	Day 30	4990.75	Day 300	10,672.95
43.28	114.94°	$3.54 \times 10^{-11}$	Day 45	5775.01	Day 330	11,003.43
72.03	107.53°	$8.60 \times 10^{-12}$	Day 60	6374.02	Day 360	11,314.08
72.54	117.56°	$1.28 \times 10^{-12}$	Day 90	7283.71		



**Figure 3.** Presentations of mesh discretization, the calculated pressure distributions at different times, and the cumulative production curve.

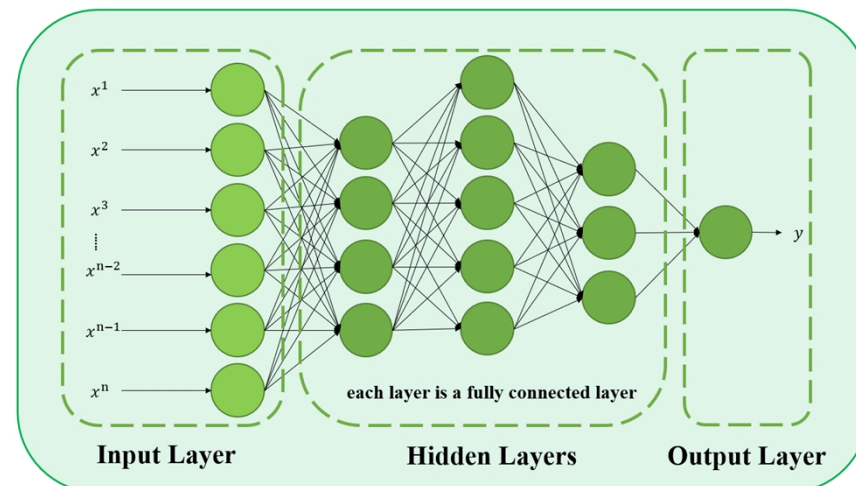
### 3. AI Methodology

In this study, three deep learning algorithms were used to construct three proxy models. These three models included MLP, regular Seq2Seq (LSTM–Attention–LSTM), and the PIED proposed in this study. MLP is widely used to construct proxy models, but MLP is not the structure specific to the sequence problem. Meanwhile, LSTM–Attention–LSTM is a common sequence-to-sequence model but does not have physical constraints integrated. In this section, the functions, structures, and principles of these three models are introduced in detail. And the training workflow of PIED is also elaborated.



### 3.1. MLP

Many studies have utilized multi-layer perceptron (MLP) to predict the production of hydraulic fracturing wells (Wang et al., 2019; Luo et al., 2018; Panja et al., 2018; Li et al., 2022) [4,30–32]. In this study, we also adopted MLP as one of our alternative proxy models. MLP consists of an input layer, hidden layers, and an output layer. Each hidden layer is a fully connected layer. A fully connected layer, also known as a dense layer, is a type of layer commonly used in neural networks. It connects every neuron from the previous layer to every neuron in the current layer, creating a fully connected network structure. The fully connected layers of the MLP are illustrated in Figure 4.



**Figure 4.** Illustration of the MLP.

The operation of the MLP is based on the forward propagation process. Input data are passed through the input layer and transmitted to the hidden layers. Upon receiving the input data, each neuron in the hidden layers performs a weighted sum and undergoes a non-linear transformation through an activation function. The transformed results are then propagated to the neurons in the next layer. The neurons in the output layer generate the final prediction.

### 3.2. Seq2Seq Architecture

#### 3.2.1. LSTM

Long short-term memory (LSTM) is a variant of recurrent neural networks (RNNs) that addresses the issue of long-term dependencies in traditional RNNs (Hochreiter et al., 1997) [33]. It provides powerful modeling capabilities for various sequence tasks. LSTM utilizes memory cells to store and update information and employs gate mechanisms to control the flow of information, effectively capturing long-term dependencies in sequences. LSTM consists of input gates, forget gates, output gates, and candidate memory cells. The input gate determines the relevance of the current input, the forget gate determines what is retained or forgotten from the previous memory, and the output gate controls the output of the memory. The candidate memory cell computes the candidate value for the current time step. The structure of a LSTM cell is depicted in Figure 5. The computation process of LSTM is as follows.

Input sequence:  $X = [x_1, x_2, \dots, x_m]$

Initial hidden state:  $h_1$

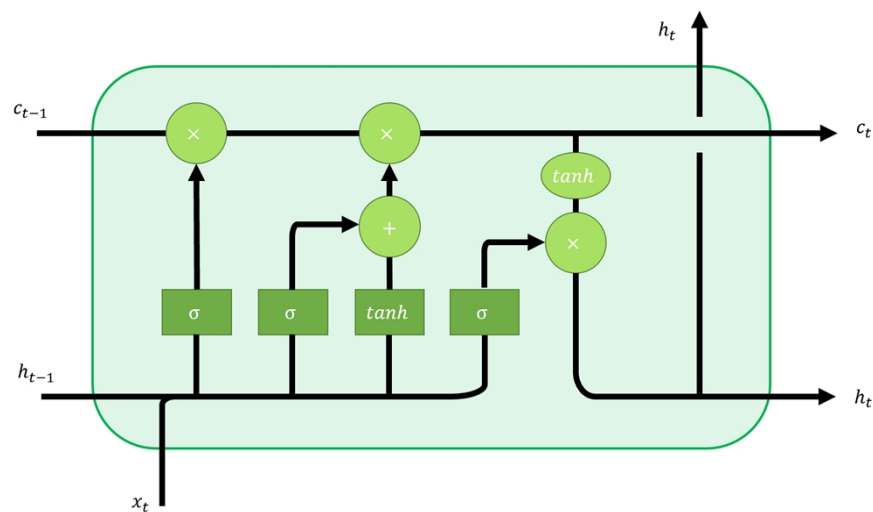
Initial cell state:  $c_1$

For each time step  $t$ ,

$y_t, h_t, c_t = LSTM(x_t, h_{t-1}, c_{t-1})$

Final hidden state at the last time step:  $h_n$

Final cell state at the last time step:  $c_n$



**Figure 5.** Illustration of the computational process of an LSTM cell at a single time step.

The LSTM (long short-term memory) uses the following formulas:

$$\text{Input gate: } i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$$

$$\text{Forget gate: } f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$$

$$\text{Output gate: } o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$\text{Candidate memory cell: } \tilde{c}_t = \tanh(W_c[h_{t-1}, x_t] + b_c)$$

$$\text{Memory cell: } c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

$$\text{Hidden state (output): } h_t = o_t \odot \tanh(c_t)$$

In the above formulas:

$\sigma$  denotes the sigmoid activation function.

$\tanh$  denotes the hyperbolic tangent activation function.

The symbol  $\odot$  represents element-wise multiplication (also known as the Hadamard product).

$W_i, W_f, W_o, W_c$  are weight matrices associated with input, forget, output, and candidate memory cells, respectively.

$b_i, b_f, b_o, b_c$  are bias vectors associated with input, forget, output, and candidate cell states, respectively.

### 3.2.2. Regular Encoder–Decoder Architecture and Attention Mechanism

#### (1) Regular Encoder–Decoder architecture

The Seq2Seq (Sequence-to-Sequence) structure is a neural network model used to convert one sequence into another (Sutskever et al., 2014) [34]. It is a special type of encoder–decoder architecture. The encoder is used to map the input sequence into a fixed-dimension vector, often referred to as a context vector. The decoder then uses the context vector to generate the target sequence. When the Seq2Seq structure is used for regression problems, it can learn the non-linear mapping relationship between the input and output sequences.

In general, the encoder utilizes LSTM as the encoder model. The encoder processes the input sequence step by step and performs recurrent computations to integrate the information from each time step into the hidden state. The hidden states at each time step are also passed through an attention module to obtain the most relevant information for the prediction. These pieces of information are then fed into the decoder structure. The decoder consists of an LSTM and a fully connected layer. It receives the feature vector  $((h_n, c_n)$  in Figure 6) generated by the encoder and the context vector  $(\alpha$  in Figure 6) attention module, and then computes the output sequence  $(h'$  in Figure 6). At each time step, the output sequence of the LSTM of the decoder is inputted into a fully connected layer of

the decoder, and the fully connected layer can calculate a production value. The Seq2Seq structure is illustrated in Figure 6.

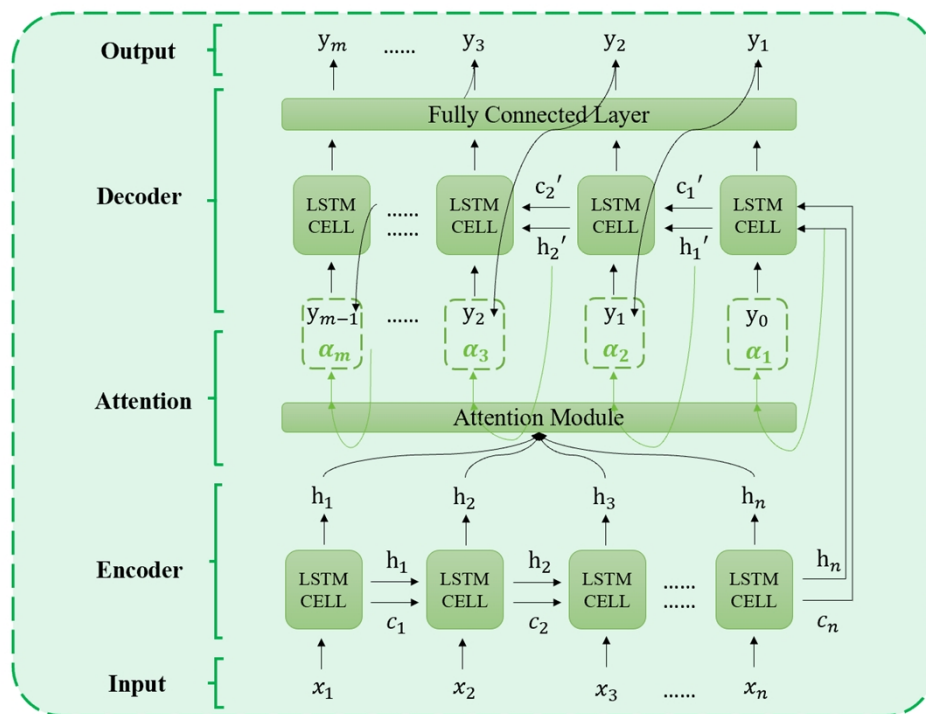


Figure 6. Illustration of a regular Seq2Seq structure.

(2) Attention mechanism

When it comes to the sequence-to-sequence (Seq2Seq) architecture, the attention mechanism is a crucial component that enables the model to selectively focus on different parts of the input sequence (in this study, the fracture information at each time step) while predicting each element of the target sequence (in this study, the predicted production at each time step).

To achieve this, the attention mechanism introduces a learnable attention model that calculates the relevance scores between the hidden states of the encoder at each time step and the hidden state of the decoder at the current time step. These scores reflect the importance of the fracture information processed at various time steps in the encoder for the current decoding step.

Specifically, the computation process of the attention mechanism is as follows:

Given the decoder’s hidden state  $h'_{t-1}$  at time step  $t - 1$  and the hidden state vectors of the encoder  $H = (h_1, h_2, \dots, h_n)$ ,

$$\begin{aligned}
 Q &= h'_{t-1}W_q \\
 K &= HW_k \\
 V &= H = (h_1, h_2, \dots, h_n) \\
 scores &= softmax(QK^T / \sqrt{n}) \\
 \alpha_t &= scores V^T
 \end{aligned}$$

Here,  $W_q$  and  $W_k$  are the learnable weight matrices used for linear transformations; ‘softmax’ is the softmax function commonly used in machine learning; and  $\alpha_t$  is the context vector.

As illustrated in Figure 6, the context vector  $\alpha_t$  is concatenated with the output of the decoder at time step  $t - 1$  and then serves as the input to the decoder at time step  $t$ . This allows the decoder to receive the most relevant fracture information that needs attention at time step  $t$ .

### 3.2.3. Physics-Informed Encoder–Decoder Architecture

For horizontal wells with non-equal intersecting hydraulic fractures, the numerical simulator considers many variables in predicting production. For example, geological factors, fracture properties and distribution, production time, etc. For a horizontal well which requires hydraulic fracturing, the geological information and information regarding entrance hole have already been determined, and the factors influencing the production are left to be considered, including fracture properties and production time. Therefore, the PIED inputs include all hydraulic fracture lengths, permeability values, and dip angles, while production times are used as intermediate inputs.

Since we have assumed that the fractures have different inclinations and lengths, there are intersections between the fractures. Different orders of fractures will lead to different intersections of fractures and thus, necessarily, different production rates. Therefore, the order of the fractures is physically meaningful to the model, and we need to treat the fractures as ordered sequential data. The predicted production is also a time series. Therefore, we chose the Seq2Seq architecture to construct the proxy model.

However, we found that the regular Seq2Seq (LSTM–Attention–LSTM) has shortcomings in solving the problem—the input of the decoder is automatically set to the output of the previous time step. This leads to the problem that the production time has to be input to the encoder together with the fracture parameters. However, in fact, in numerical simulations, the properties of the fractures were not related to the production time, so it is not reasonable to couple the time information with the physical information of the fractures in the encoder. Since the encoder will extract the information of the input, if the production time is input into the encoder structure, then the time information received by the decoder will not be complete.

Based on the considerations mentioned above, PIED adopts a modified Seq2Seq structure, which causes the decoder to have a constant intermediate input by manually defining the input for each time step when constructing the decoder. The intermediate input is the production time used in the numerical simulator. The PIED consists of an encoder and a decoder. The encoder consists of an LSTM combined with an attention module, while the decoder consists of an LSTM and a fully connected layer. The input of the encoder is a sequence with a shape of  $3 \times n$  consisting of the length, permeability, and dip angle of  $n$  fractures (10 in this study). The intermediate input of the decoder is a sequence consisting of 15 numerical type variables. The structure of the PIED is shown in Figure 7.

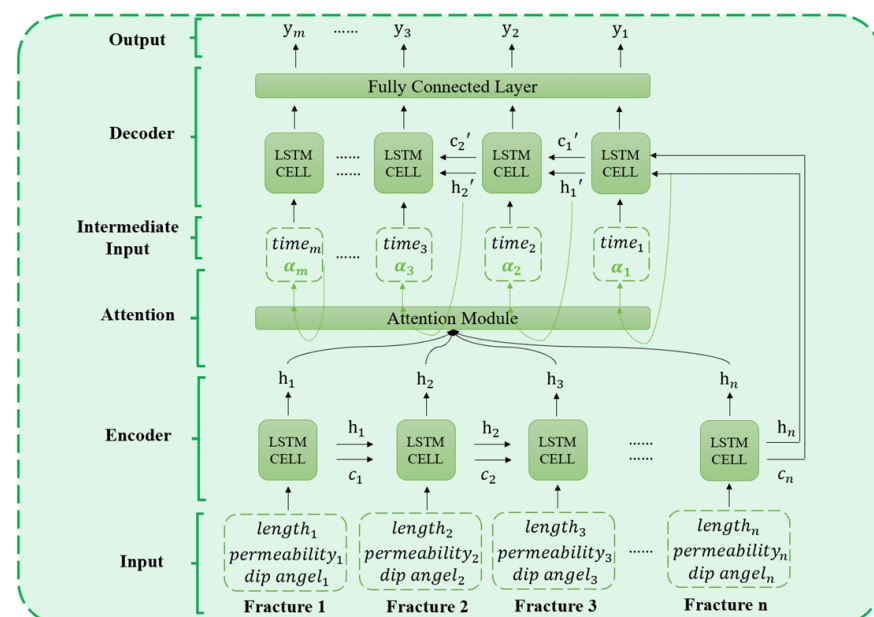


Figure 7. Illustration of the computational process of the PIED.

The encoder is able to extract the input (i.e., the sequence of  $3 \times n$  fractures) to feature vectors (i.e., the hidden states and cell states of the encoder at the last time step), and by introducing the attention module, the proxy model can automatically extract the fracture information, which has an impact on the decoder from all the hidden states of the encoder, thus reducing the risk of overfitting. The LSTM of the decoder combines the feature vector output by the encoder, the production time series (i.e., the intermediate input), and the vectors output by the attention module to output a vector in each time step. And after each time step, the fully connected layer inputs the vector to predict a production. In detail, the hidden state and cell state of the encoder at the last time step will be used as the initial hidden state and initial cell state of the decoder's LSTM. And at each time step of the decoder's LSTM, the vector output from the attention module is combined with the production time and then input into the LSTM. Thus, the decoder will be constrained by physical information regarding the fractures in predicting the production. We also compare the PIED with the regular LSTM–Attention–LSTM in Section 4 to demonstrate the improvement in performance as a result of this modification.

In addition, we chose LSTM as the decoder because the output production is a time series and LSTM, as a model specialized in sequential problems, can fully consider the effect of the production in the earlier period on the production in the later period. The plot of correlation coefficients among the variables in one case is shown in Figure 8, showing a high correlation between the productions at different production times. We also show the superiority of LSTM in this problem by comparing it with the widely used MLP proxy model in Section 4.

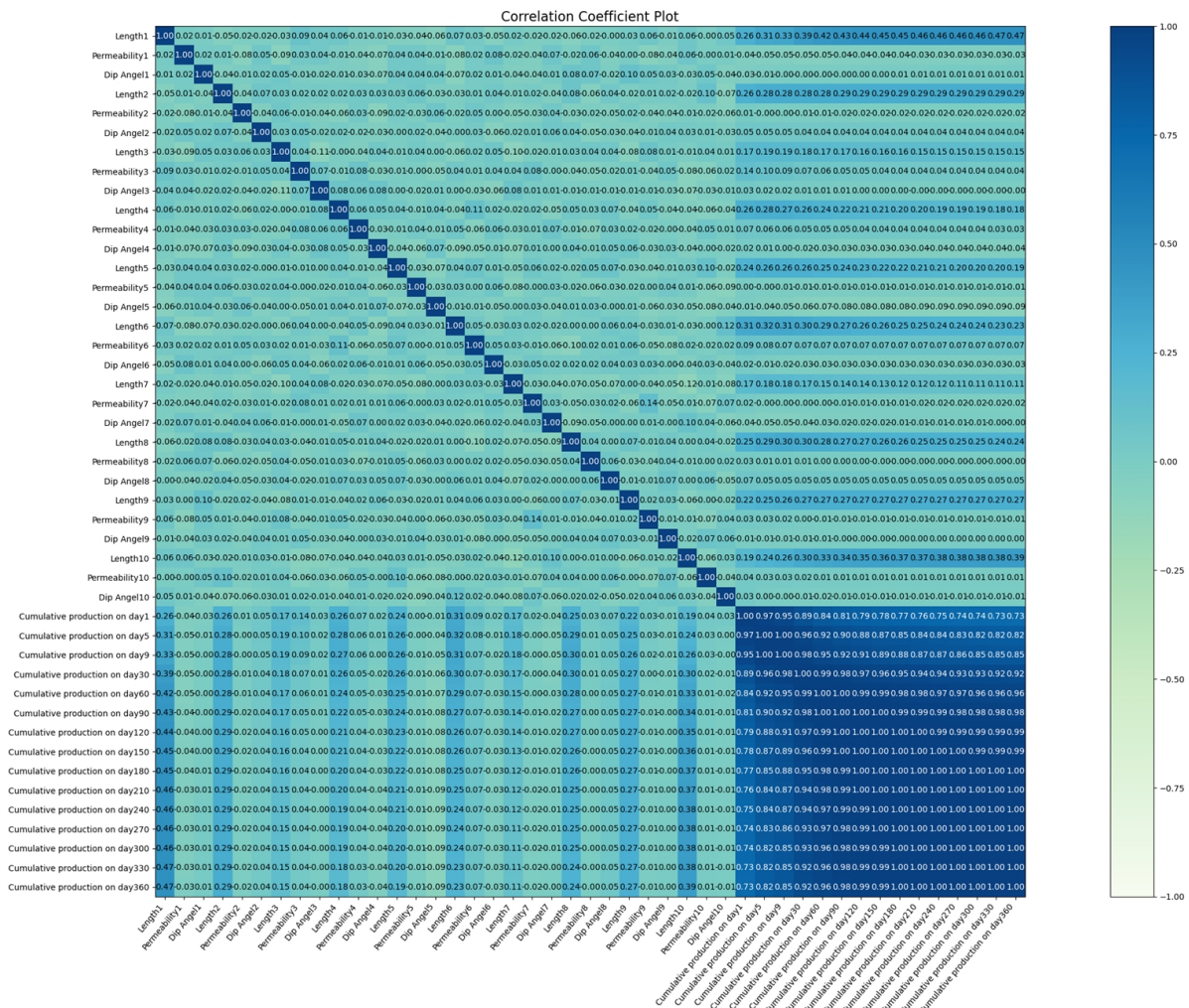


Figure 8. The plot of correlation coefficients among the variables in a case study.

### 3.3. The Training Workflow of PIED

A schematic diagram of the training process of the PIED is shown in Figure 9.

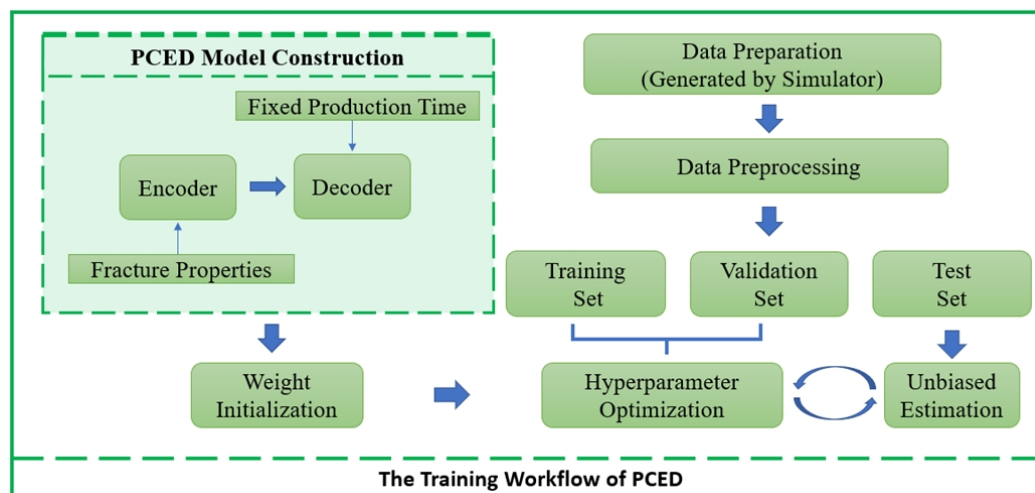


Figure 9. The training workflow of the PIED.

#### 3.3.1. Data Preparation

The dataset used to train the proxy models is generated by the numerical simulator mentioned in Section 2. For the horizontal wells that are required for production prediction, some static parameters such as the geological parameters (e.g., permeability and porosity of the matrix) and parameters of well completion (e.g., perforated interval) need to be determined at first. Then, the number of hydraulic fractures as well as the ranges of values of fracture parameters is set according to the engineering background and a series of time points at which the production need to be predicted are determined. After that, the fracture length, permeability, and dip angle are randomly generated using a uniform distribution. Finally, the generated fracture parameters and the static parameters determined previously are input into the numerical simulator. Finally, the generated fracture parameters and the previously determined static parameters are input into the numerical simulator. Finally, the numerical simulator predicts the production at each time point. As an example, a sample of 10 fractures was generated with this method and is presented in Table 3.

Table 3. A sample generated by the numerical simulation.

A Sample Generated by the Numerical Simulator			
	Length	Permeability	Dip angel (radian)
Fracture1	9.173320731	$3.33587 \times 10^{-13}$	1.737934557
Fracture2	52.73803329	$4.45321 \times 10^{-13}$	1.590268905
Fracture3	7.970179625	$1.1239 \times 10^{-8}$	2.066094122
Fracture4	10.00118252	$8.87251 \times 10^{-12}$	1.726819853
Fracture5	41.51548897	$2.33735 \times 10^{-9}$	1.88530177
Fracture6	11.7711018	$1.84973 \times 10^{-10}$	1.522413401
Fracture7	62.27039877	$3.97928 \times 10^{-11}$	1.499996875
Fracture8	62.22829645	$2.58495 \times 10^{-13}$	1.911464137
Fracture9	55.57077147	$4.9175 \times 10^{-10}$	1.134606937
Fracture10	15.49058097	$3.71595 \times 10^{-13}$	1.186653904
PRODUCTION Series	[767.3, 1580.8, 2030.5, 3271.0, 4266.5, 4974.3, 5537.9, 6013.7, 6430.8, 6805.8, 7149.1, 7467.8, 7766.5, 8048.6, 8316.7]		Generated by the numerical simulator (in Section 2)

After the dataset is generated by the numerical simulator, the dataset will be randomly split into a training set, a validation set, and a test set at a ratio of 8:1:1.

### 3.3.2. Data Preprocessing

Normalization is a common method of data preprocessing. Normalization scales the data so that different features have similar scales, thus removing the impact of scales on the data and helping to speed up the convergence process of the model. This reduces the sensitivity of the model to noise or outliers and improves the robustness and generalization ability of the model.

Min–Max normalization has shown favorable results in data preprocessing. This technique employs a linear transformation on the original data, mapping the minimum value to 0 and the maximum value to 1, while scaling the remaining data proportionally based on their relative positions. The advantages of Min–Max normalization are preserving the original distribution shape of the data and ensuring a linear mapping of the data within the specified range. Min–Max normalization is calculated as follows:

$$x_{normalized} = (x - \min(x)) / (\max(x) - \min(x))$$

In this representation, “ $x_{normalized}$ ” represents the normalized value, “ $x$ ” represents the original value, “ $\min(x)$ ” represents the minimum value in the dataset, and “ $\max(x)$ ” represents the maximum value in the dataset.

### 3.3.3. Model Construction and Weight Initialization

The coding of the PIED and its forward propagation steps can be done using PyTorch 1.13.0 based on Python 3.9.

Before the forward propagation, Xavier initialization (Glorot et al., 2010) [35] is used to ensure that the variances of the activation values and gradients remain roughly the same across different layers, which helps effectively propagate gradients during the training process. For the weight matrix of an LSTM unit, assuming the input dimension is  $n_{in}$  and the output dimension is  $n_{out}$ , during Xavier initialization, the elements of the weight matrix are still randomly sampled from a uniform distribution, and the range can be calculated using the following formula:

$$a = \sqrt{\frac{6}{(n_{in} + n_{out})}}$$

$$w = \text{uniform}(-a, a)$$

Here,  $n_{in}$  is the input dimension and  $n_{out}$  is the output dimension.

For LSTM, since it has complex structures such as input gates, forget gates, and output gates, each gate has its own weight matrix. During Xavier initialization, the weight matrices of each gate should be initialized based on the corresponding input and output dimensions. This ensures that the weight initialization range for each gate is appropriate and facilitates gradient propagation within the LSTM unit.

During the backward propagation process, the Adam optimization algorithm (Kingma et al., 2014) [36] is used. The Adam algorithm combines the characteristics of adaptive learning rate and momentum, enabling effective learning rate adjustment and fast convergence.

### 3.3.4. Hyperparameter Optimization

Grid Search is a commonly used hyperparameter optimization method that aims to determine the best combination of hyperparameters to optimize model performance. The PIED hyperparameter space consists of the learning rate, the hidden size of LSTM, and the batch size. For a more comprehensive search, additional parameters such as weight decay, the number of fully connected layers, and the number of neurons can also be considered.

Once a promising combination of hyperparameters is found, learning curves can be plotted for further fine-tuning. In this study, the training curve vs. validation curve method was employed to evaluate model overfitting by simultaneously observing the learning curves of the training and testing datasets. Typically, the x-axis represents the number of iterations or epochs of training, while the y-axis represents the model’s loss or accuracy.

When the training loss continues to decrease and the testing loss starts to increase, this indicates the model is starting to overfit.

### 3.3.5. Model Evaluation

When solving regression problems, three commonly used model evaluation metrics include mean absolute error (MAE), mean squared error (MSE), and root mean squared error (RMSE). These model evaluation metrics can be calculated as follows:

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}$$

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

In the above formula,  $y_i$  represents the true values (i.e., the production series calculated by the numerical simulator mentioned in Section 2),  $\hat{y}_i$  represents the predicted values by the DL proxy model, and  $n$  represents the number of samples.

## 4. Case Study

### 4.1. Data Preparation and Preprocessing

Based on the data generation method proposed in Section 3 and the numerical simulator mentioned in Section 2, a dataset was created for predicting the cumulative production of horizontal wells with unequal-length and intersecting hydraulic fractures on a two-dimensional plane during depletion. Under the given geological static parameters and perforation spacing conditions, we randomly generated 10 hydraulic fractures with different lengths, permeabilities, and dip angles, as shown in Figure 10. The production was then computed using the numerical simulator for the following time intervals: 1, 5, 9, 30, 60, 90, 120, 150, 180, 210, 240, 270, 300, 330, and 360 days. The lengths, permeabilities, and dip angles of these hydraulic fractures are uniformly distributed within specified ranges. In the end, we generated 500 such samples as the dataset, and a sample is presented as an example in Table 3. The distributions of each feature in the dataset are visualized in Figure 11. Specifically, the range of fracture lengths is 5~75 m, the range of permeabilities is  $1 \times 10^{-13} \sim 1 \times 10^{-8}$  D, and the range of inclinations is  $\pi/6 \sim 2\pi/3$ . After applying Min–Max normalization, the dataset was split into training, validation, and testing sets at a ratio of 8:1:1.

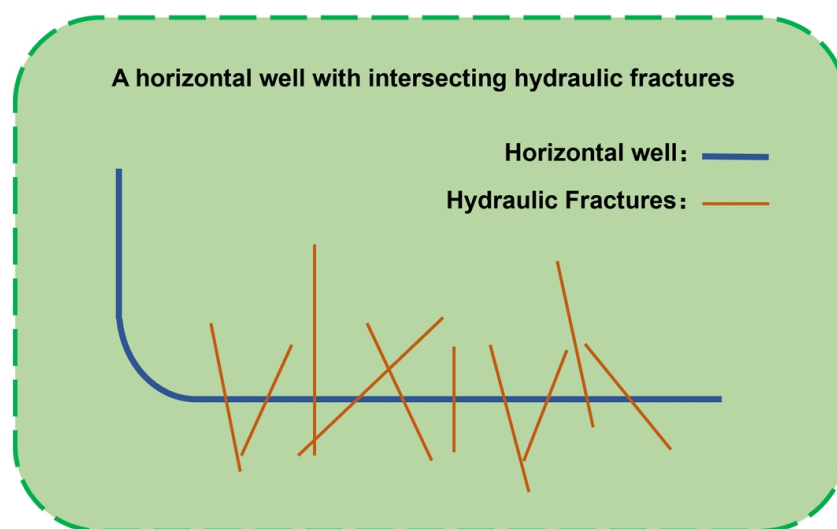


Figure 10. Illustration of a sample of 10 fractures from the case study.



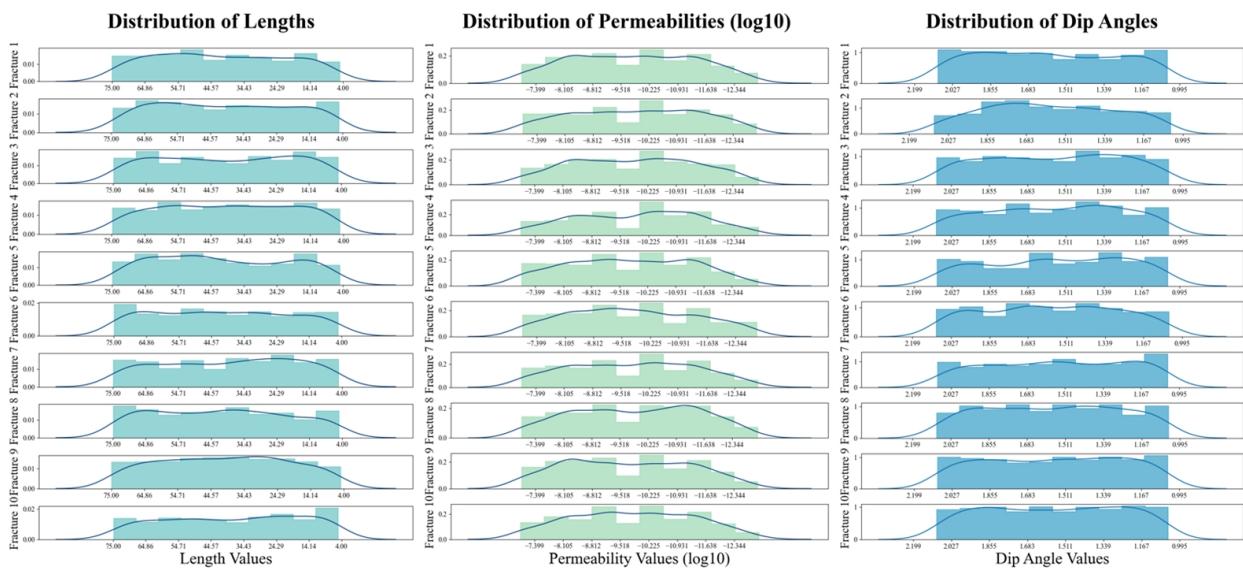


Figure 11. The distribution of the lengths, permeabilities, and dip angles of all the fractures in the dataset.

#### 4.2. Configuration of Proxy Models and Hyperparameter Optimization

Based on the “training workflow” described in Section 3, MLP, LSTM–LSTM, and the PIED were constructed and optimized with hyperparameters. The hyperparameters used for training the three surrogate models are presented in Table 4.

Table 4. The hyperparameters of the proxy models we trained in the case study.

Proxy Models	Hidden Size of LSTM	Fully Connected Layers	Dropout	Activation Function	BatchNorm
PIED	Encoder: 13	Encoder: 0	—	—	—
	Decoder: 13	Decoder: 1 (neurons: 13)	—	—	Not Applied
Regular LSTM-Attention-LSTM	Encoder: 12	Encoder: 0	—	—	—
	Decoder: 12	Decoder: 1 (neurons: 12)	—	—	Not Applied
MLP	—	4 (neurons: 28, 24, 24, 16)	0.2, 0.1, 0, 0	Leaky Relu	Applied

#### 4.3. Evaluation of the Proxy Models

The performance of the three proxy models were evaluated based on the metrics mentioned in the ‘model evaluation’ part of Section 3. By predicting the data from the test set, the MSE, MAE, and RMSE values of these three models are shown in Figure 12. The PIED achieved the best results regarding MSE, MAE, and RMSE, while regular LSTM–Attention–LSTM outperformed MLP in all three metrics. The predictions of the three proxy models for the test set are shown in Figure 13.

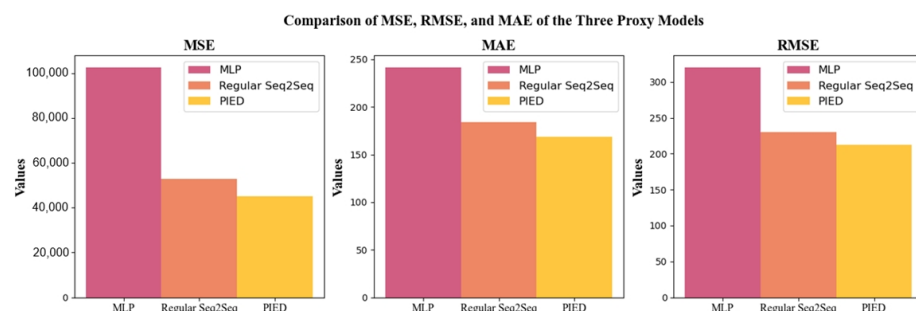


Figure 12. The comparison of MSE, RMSE and MAE of the three proxy models.

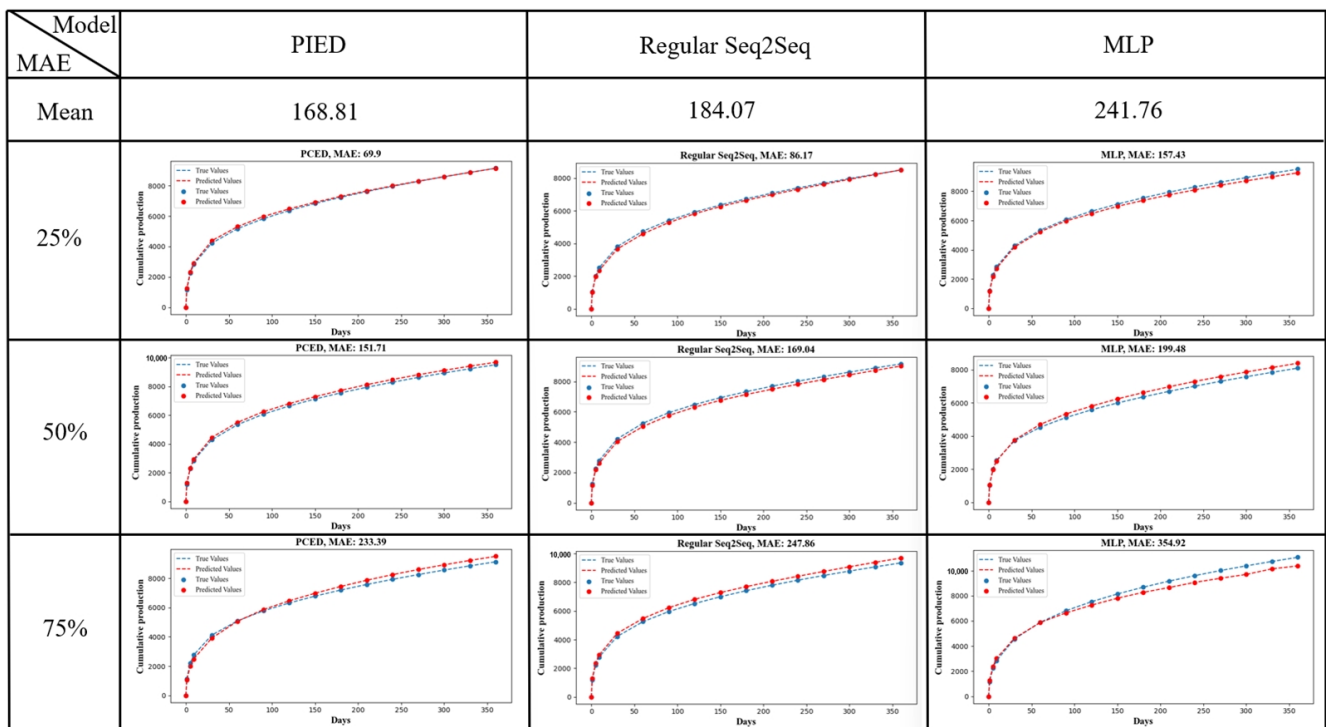


Figure 13. Illustration of the predictive performance of the three agent models on the test set.

#### 4.4. Contribution of Physics Information and Seq2Seq Structure

##### (1) Superiority of the Seq2Seq Structure:

In terms of model performance, compared to MLP’s MSE, LSTM–Attention–LSTM’s MSE decreased by 48.4%. Seq2Seq architecture outperformed MLP in predicting the production sequence.

When analyzing the model structures, we first employed a multi-layer perceptron (MLP) as the decoder and constructed an LSTM–MLP architecture for comparative experimentation in the test set. The comparative results of MLP, LSTM–MLP, and LSTM–Attention–LSTM are illustrated in Figure 14. The superior performance of LSTM–MLP over MLP demonstrates that the LSTM structure is better equipped to handle fracture information. Given that fractures intersect and their positions and order affect production, leveraging LSTM to process fissures as sequential features enhances its performance. The fact that the performance of LSTM–Attention–LSTM surpassed that of LSTM–MLP further validates its stronger performance in predicting production sequences using LSTM.

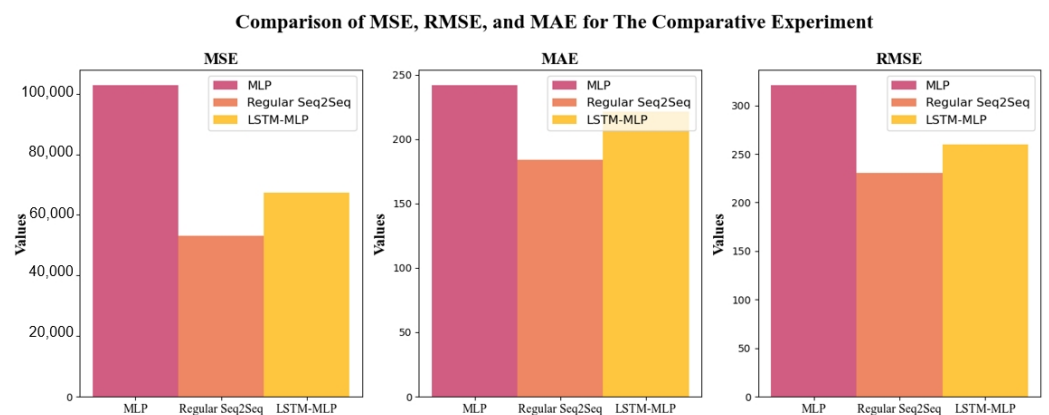


Figure 14. The comparison of the MSE, RMSE, and MAE values of MLP, regular Seq2Seq, and the LSTM–MLP.

From the perspective of the number of internal parameters (i.e., weights and biases), in this case, MLP has 2557 internal parameters, while LSTM–Attention–LSTM has 3020 internal parameters. This is one of the reasons why MLP exhibits more severe overfitting compared to LSTM–Attention–LSTM. Since MLP treats the input fractures as discrete input features, while LSTM–Attention–LSTM treats them as a sequence, the internal parameters of MLP increases with the number of fractures, whereas that of LSTM–Attention–LSTM does not. In this case, where 10 fractures have 30 fracture variables and 15 time variables, the input dimension is high. This results in MLP having more internal parameters compared to LSTM–Attention–LSTM. Consequently, training MLP requires more data compared to training LSTM–Attention–LSTM, and it also implies that constructing the MLP model incurs more costs to generate additional samples.

## (2) Improvement of Seq2Seq Performance using Physical Information:

Compared to that of LSTM–Attention–LSTM, the PIED’s MSE decreased by 11.18%. It is evident that by separately inputting the production time series into the decoder, the PIED exhibits superior performance.

From the perspective of production calculation, the geometry attributes of fractures and the set production time are independent in the XFEM numerical simulator when it is calculating the production. the geometric parameters of the fractures remain unchanged, and therefore, PIED has an encoder structure that can independently process all the fracture information. Meanwhile, when it comes to calculating production, the time information will be coupled with fracture information in the decoder. However, in the regular Seq2Seq model, time information and fracture information are input together from the beginning.

From the viewpoint of model structure, the regular LSTM exhibits error accumulation, while the PIED’s structure mitigates error accumulation. Specifically, for the regular LSTM, it takes the output of the previous time step as the input for the current time step. As the output of the previous time step contains errors, each LSTM at every time step is provided with information containing accumulated errors. Therefore, the more time steps, the more error accumulation in LSTM. However, in the PIED, at each time step, the input of the decoder’s LSTM are the predetermined static values from the production time series and the vectors outputted by the attention module, so that the accumulated error in the PIED is reduced.

## 5. Conclusions

In this study, the PIED, a new physics-informed proxy model of numerical simulator for fractured horizontal well production prediction, is proposed, and the Seq2Seq architecture is initially used to do this. The encoder structure and attention algorithm successfully extract physical information regarding intersecting hydraulic fractures, and the modified decoder structure succeeds in predicting production series within the constraints of production time and the fracture information outputted by the encoder and the attention module. We also proved that the intersecting hydraulic fractures processed as the series type data by the LSTM can enhance the performance of the proxy model. Finally, according to the case study, the performance of the Seq2Seq architecture significantly surpassed the commonly used MLP, highlighting its superiority. Furthermore, the performance of the PIED exhibited further improvement upon the foundation of Seq2Seq.

In the case study, where the feature dimensionality was high, with 30 variables forming a sequence of size  $3 \times 10$  as the input and 15 fixed time values as the intermediate input for the decoder, the PIED exhibited satisfactory performance in handling high-dimensional features. The MAE of the PIED was 168.8, which is equal to 2.7% bias of the true cumulative production values. Compared to the MSE values of MLP and LSTM–Attention–LSTM, the MSE of PIED was 56.0% and 11.8% less, respectively. The PIED model demonstrates its stronger generalization ability and higher accuracy. Only 500 samples need to be generated over 3 days to build datasets, so that PIED exhibits practicality as it can be trained and fine-tuned on small datasets, achieving satisfactory performance.

## 6. Discussion

The physics-informed encoder–decoder (PIED) model, while capable of achieving strong performance on smaller datasets, still exhibits some limitations. Firstly, the PIED currently incorporates only two types of physical information, namely, the geometric features of fractures and production time. However, incorporating more physical information, such as the geological conditions of the matrix, should be considered. Secondly, while the PIED performed well in the case of 10 hydraulic fractures, its encoder’s ability to handle a much larger number of input variables, say a hundred fractures, needs to be validated.

Fortunately, the aforementioned limitations of the PIED can be addressed. To incorporate additional types of physical information, other network structures such as fully connected layers can be employed to map the information into vectors, which can then be used as input or hidden states in the PIED. Moreover, when dealing with a higher number of fractures, the encoder can be equipped with a self-attention mechanism to further mitigate overfitting.

**Author Contributions:** Data preparation and numerical simulations, Y.X.; proxy models, AI methodology, and case Study, T.J.; AI methodology and case study, H.J. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was financially supported by the National Natural Science Foundation of China (Grant NO. U19B6003-05).

**Data Availability Statement:** Not applicable.

**Acknowledgments:** The financial support provided by the National Natural Science Foundation of China is highly appreciated.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Wang, H.; Qiao, L.; Lu, S.; Chen, F.; Fang, Z.; He, X.; Zhang, J.; He, T. A Novel Shale Gas Production Prediction Model Based on Machine Learning and Its Application in Optimization of Multistage Fractured Horizontal Wells. *Front. Earth Sci.* **2021**, *9*, 675–689. [[CrossRef](#)]
2. Wang, T.; Wang, Q.; Shi, J.; Zhang, W.; Ren, W.; Wang, H.; Tian, S. Productivity Prediction of Fractured Horizontal Well in Shale Gas Reservoirs with Machine Learning Algorithms. *Appl. Sci.* **2021**, *11*, 12064. [[CrossRef](#)]
3. Xue, H.; Malpani, R.; Agrawal, S.; Bukovac, T.; Mahesh, A.L.; Judd, T. Fast-Track Completion Decision Through Ensemble-Based Machine Learning. In Proceedings of the SPE Reservoir Characterisation and Simulation Conference and Exhibition, Abu Dhabi, United Arab Emirates, 17–19 September 2019. [[CrossRef](#)]
4. Lizhe, L.; Fujian, Z.; You, Z. The prediction and optimization of Hydraulic fracturing by integrating the numerical simulation and the machine learning methods. *Energy Rep.* **2022**, *8*, 15338–15349. [[CrossRef](#)]
5. Wang, B.; Han, G.; Lu, X.; Ma, H.; Zhu, Z.; Liang, X. The integrated geosciences and engineering production prediction in tight reservoir based on deep learning. *Geoenergy Sci. Eng.* **2023**, *223*, 211571. [[CrossRef](#)]
6. Krishnapriyan, A.; Gholami, A.; Zhe, S.; Kirby, R.; Mahoney, M.W. Characterizing possible failure modes in physics-informed neural networks. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 26548–26560.
7. Cai, S.; Mao, Z.; Wang, Z.; Yin, M.; Karniadakis, G.E. Physics-informed neural networks (PINNs) for fluid mechanics: A review. *Acta Mech. Sin.* **2021**, *37*, 1727–1738. [[CrossRef](#)]
8. Cursi, J.S.D.; Koscianski, A. Physically constrained neural network models for simulation. In Proceedings of the Advances and Innovations in Systems, Computing Sciences and Software Engineering, Bridgeport, CT, USA, 3–12 December 2007; Springer: Amsterdam, The Netherlands, 2007.
9. Cuomo, S.; Di Cola, V.S.; Giampaolo, F.; Rozza, G.; Raissi, M.; Piccialli, F. Scientific machine learning through physics-informed neural networks: Where we are and what’s next. *J. Sci. Comput.* **2022**, *92*, 88. [[CrossRef](#)]
10. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [[CrossRef](#)]
11. Zhu, Y.; Zabaraz, N.; Koutsourelakis, P.S.; Perdikaris, P. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. *J. Comput. Phys.* **2019**, *394*, 56–81. [[CrossRef](#)]
12. Li, X.; Ma, X.; Xiao, F.; Xiao, C.; Wang, F.; Zhang, S. A physics-constrained long-term production prediction method for multiple fractured wells using deep learning. *J. Petrol. Sci. Eng.* **2022**, *217*, 110844. [[CrossRef](#)]

13. Guevara, J.; Zdrozny, B.; Buoro, A.; Lu, L.; Tolle, J.; Limbeck, J.W.; Hohl, D. A machine-learning methodology using domain-knowledge constraints for well-data integration and well-production prediction. *SPE Reserv. Eval. Eng.* **2019**, *22*, 1185–1200. [[CrossRef](#)]
14. Qu, H.Y.; Zhang, J.L.; Zhou, F.J.; Peng, Y.; Pan, Z.J.; Wu, X.Y. Evaluation of hydraulic fracturing of horizontal wells in tight reservoirs based on the deep neural network with physical constraints. *Pet. Sci.* **2023**, *20*, 1129–1141. [[CrossRef](#)]
15. Yang, R.; Qin, X.; Liu, W.; Huang, Z.; Shi, Y.; Pang, Z.; Wang, T. A physics-constrained data-driven workflow for predicting Coalbed methane well production using artificial neural network. *SPE J.* **2022**, *27*, 1531–1552. [[CrossRef](#)]
16. Wang, N.; Chang, H.; Zhang, D. Surrogate and inverse modeling for two-phase flow in porous media via theory-guided convolutional neural network. *J. Comput. Phys.* **2022**, *466*, 111419. [[CrossRef](#)]
17. Cornelio, J.; Mohd Razak, S.; Cho, Y.; Liu, H.H.; Vaidya, R.; Jafarpour, B. Neural Network-Assisted Clustering for Improved Production Predictions in Unconventional Reservoirs. In Proceedings of the SPE Western Regional Meeting, Anchorage, AK, USA, 22–25 May 2023.
18. Razak, S.M.; Cornelio, J.; Cho, Y.; Liu, H.H.; Vaidya, R.; Jafarpour, B. Embedding physical flow functions into deep learning predictive models for improved production forecasting. In Proceedings of the Unconventional Resources Technology Conference, Houston, TX, USA, 20–22 June 2022.
19. Baca, R.G.; Arnett, R.C.; Langford, D.W. Modelling fluid flow in fractured-porous rock masses by finite-element techniques. *Int. J. Numer. Methods Fluids* **1984**, *4*, 337–348. [[CrossRef](#)]
20. Xia, Y.; Jin, Y.; Oswald, J.; Chen, M.; Chen, K. Extended finite element modeling of production from a reservoir embedded with an arbitrary fracture network. *Int. J. Numer. Methods Fluids* **2018**, *86*, 329–345. [[CrossRef](#)]
21. Jin, Y.; Chen, K.; Chen, M. An asymptotic solution for fluid production from an elliptical hydraulic fracture at early-times. *Mech. Res. Commun.* **2015**, *63*, 48–53. [[CrossRef](#)]
22. Oswald, J.; Gracie, R.; Khare, R.; Belytschko, T. An extended finite element method for dislocations in complex geometries: Thin films and nanotubes. *Comput. Methods Appl. Mech. Eng.* **2009**, *198*, 1872–1886. [[CrossRef](#)]
23. Karimi-Fard, M.; Durlofsky, L.J.; Aziz, K. An efficient discrete fracture model applicable for general purpose reservoir simulators. *SPE J.* **2004**, *9*, 227–236. [[CrossRef](#)]
24. Reichenberger, V.; Jakobs, H.; Bastian, P.; Helmig, R. A mixed-dimensional finite volume method for two-phase flow in fractured porous media. *Adv. Water Resour.* **2006**, *29*, 1020–1036. [[CrossRef](#)]
25. Noorishad, J.; Mehran, M. An upstream finite element method for solution of transient transport equation in fractured porous media. *Water Resour. Res.* **1982**, *18*, 588–596. [[CrossRef](#)]
26. Xia, Y.; Jin, Y.; Chen, M.; Chen, K. An enriched approach for modeling multiscale discrete-fracture/matrix interaction for unconventional-reservoir simulations. *SPE J.* **2019**, *24*, 349–374. [[CrossRef](#)]
27. Karimi-Fard, M.; Firoozabadi, A. Numerical simulation of water injection in fractured media using the discrete-fractured model and the galerkin method. *SPE Res. Eval. Eng.* **2003**, *6*, 117–126. [[CrossRef](#)]
28. Melenk, J.M.; Babuška, I. The partition of unity finite element method: Basic theory and applications. *Comput. Methods Appl. Mech. Eng.* **1996**, *139*, 289–314. [[CrossRef](#)]
29. Berrone, S.; Pieraccini, S.; Scialò, S. On simulations of discrete fracture network flows with an optimization-based extended finite element method. *SIAM J. Sci. Comput.* **2013**, *35*, A908–A935. [[CrossRef](#)]
30. Wang, S.; Chen, Z.; Chen, S. Applicability of deep neural networks on production forecasting in Bakken shale reservoirs. *J. Petrol. Sci. Eng.* **2019**, *179*, 112–125. [[CrossRef](#)]
31. Luo, G.; Tian, Y.; Bychina, M. Production optimization using machine learning in Bakken shale. In Proceedings of the Unconventional Resources Technology Conference, Houston, TX, USA, 23–25 July 2018; Society of Exploration Geophysicists: Tulsa, OK, USA; American Association of Petroleum Geologists: Tulsa, OK, USA; Society of Petroleum Engineers: Richardson, TX, USA, 2018.
32. Panja, P.; Velasco, R.; Pathak, M.; Deo, M. Application of artificial intelligence to forecast hydrocarbon production from shales. *Petroleum* **2018**, *4*, 75–89. [[CrossRef](#)]
33. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)]
34. Sutskever, I.; Vinyals, O.; Le, Q.V. Sequence to sequence learning with neural networks. *Adv. Neural Inf. Process. Syst.* **2014**, *2*, 3104–3112.
35. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Chia Laguna Resort, Sardinia, Italy, 13–15 March 2010.
36. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the International Conference for Learning Representations, San Diego, CA, USA, 7–9 May 2015.

**Disclaimer/Publisher’s Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.