

# Solving the Reaction-Diffusion equation based on analytical methods and deep learning algorithm; the Case study of sulfate attack to concrete

Amin Karimi Monsefi

Computer Science and Engineering  
Shahid Beheshti University  
Tehran, Iran  
a\_karimimonsefi@sbu.ac.ir

Rana Bakhtiyarzade

Metallurgy and Materials Engineering  
Iran University of Science and Technology  
Tehran, Iran  
rana\_bakhtiyarzade@metaleng.iust.ac.ir

**Abstract**—The reaction-diffusion equation is one of the cornerstones equations in applied science and engineering. In the present study, a deep neural network has been trained in order to predict the solution of the equation with different coefficients using the numerical solution of this equation and the utility of deep learning. Analytical solution of the Reaction-Diffusion equation also has been conducted by taking advantage of Danckwerts method. The accuracy of deep learning results was compared with the analytical solutions. In order to decrease the learning time and to find out similar equations solutions, such as pure diffusion and pure reaction, dimensional analysis technique has been performed. It was demonstrated that deep learning can accurately estimate the Partial Differential Equations solution in the case of the reaction-diffusion equation with a constant coefficient.

**Index Terms**—Deep Learning, Partial Differential Equations(PDEs), Reaction-Diffusion, Danckwerts Method

## I. INTRODUCTION

Partial Differential Equations (PDEs) have always played a significant role in mathematical modeling and simulation, which are widely used in physics, engineering and economics [1], [2]. A wide range of physical phenomena are modelled by second-order PDEs with constant coefficients, and representing precise solutions for these kinds of equations is an important part of applied mathematics. Although analytical and numerical methods have been widely developed for solving PDEs, each one has its own advantages and drawbacks. While analytical methods lead to precise solutions, they are not applicable for most of PDEs, and by any changes in boundary and initial conditions type, they become useless. On the other hand, numerical methods enable scientists to solve complex PDE problems, but these methods are mesh dependent and have high computational costs, and also their solutions are not reliable before validation with analytical solutions or experimental results. In order to reach a fast and accurate technique, which utilizes the advantages of both numerical and analytical methods, a mesh-free tool is needed. Consequently, an intelligent method which enable us to comprehend the analytical solutions and generalize them for complex geometries is the best choice.

Deep learning is a part of artificial intelligence which is quite useful and provide several advances in finding the solution of high dimensional data problems. In recent years several pieces of research have demonstrated significant success in artificial intelligence [3]–[12]. Although the idea of using the neural network is an old idea, in recent years researches show that a wide variety of excellent modeling of complicated data sets were established using the mentioned method [13]. Deep learning methods as a representation learning method use several levels, and each level uses non-linear schemes to representation at a higher level. By using enough transformations, complicated functions also be learned. It is noticeable that in deep learning method all of the discussed levels are designed by computer itself [4], [14].

In this work, a deep neural network algorithm has been trained by using the numerical solution of a specific PDE in finite intervals. The purpose of this network is to learn the behaviour of the equation for predicting the values of the solution for whole domain. By taking the advantage of the analytical solution, error and the accuracy of deep learning result was calculated.

Reaction-Diffusion equation is one of the most famous PDEs in engineering problems which is used as a case study in this paper. This equation is widely used for prediction of sulfate concentration in concrete during the sulfate attack process [15]–[17]. The one-dimensional Reaction-Diffusion equation with Dirichlet boundary condition has been solved analytically on symmetric line. In order to solve the analytical solution, firstly the pure diffusion equation solved, then based on danckwerts transformation the reaction-diffusion solution would be extracted from the pure Diffusion equation solution [18]. The Diffusion and the Rate of Reaction coefficients for the case of sulfate attack in concrete have assumed based on the Zuo et al work [15]. The numerical solution of the discussed equation has also extracted by the second-order algorithm of finite difference method. A deep neural network with several main hidden layers designed to learn the behavior of the equation from the numerical results, and all the influential parameters on the solution of the equation

classified into different categories, and considered as a separate input parameter for the deep learning algorithm. Both the solution and parameters were used as feed for the first layer of the algorithm and trained the neural network to predict the solution of the equation with logical possible parameters for the problem. In order to have the better judgment about the solution, a practical case study of sulfate attack which was modeled by Zuo et al [15] was used, and all the coefficients and boundary and the initial conditions were set similar to their research. The results which have calculated by the deep learning algorithm has compared to analytical results, and by using dimensional analysis methods, the solutions of pure reaction and pure diffusion was also extracted from deep learning results. The deep learning based solution is quite similar to the analytical methods with low error and high accuracy.

## II. METHODOLOGY

### A. Governing Equations

The governing equation for all the transportation phenomena including sulfate attack can be expressed as the following equation [19]:

$$\frac{\partial \rho \phi}{\partial t} + \nabla(\rho \vec{v} \phi) = \nabla(\Gamma_\phi \nabla \phi) + S_\phi \quad (1)$$

where  $\phi$  is the generalized scalar,  $\Gamma_\phi$  is global diffusion coefficient,  $S_\phi$  is the source term and,  $\vec{v}$  and  $\rho$  are the velocity and density of the fluid respectively.

It is assumed that there is no fluid in the case of sulfate attack, as a result, the convection term of the governing Eq.1 is omitted from the main equation. so it will be simplified as follows:

$$\frac{\partial \phi}{\partial t} = \nabla(\Gamma_\phi \nabla \phi) + S_\phi \quad (2)$$

Zuo [15] has reformed the governing equation using sulfate attack phenomena parameters which are represented in the following table:

TABLE I  
COEFFICIENTS OF EQUATION.

Equation	$S_\phi$	$\phi$	$\Gamma_\phi$
reaction_diffusion	$\frac{\partial C_d}{\partial t}$	$C$	$D_e$

Eventually the transport equation considering boundary and initial conditions and with the assumption of one-dimensional diffusion, the corresponding equation will be reformed to the form of Eq.3:

$$x = \begin{cases} \frac{\partial C}{\partial t} = \frac{\partial(D_e(x,t) \frac{\partial C}{\partial x})}{\partial x} + \frac{\partial C_d}{\partial t} \\ C(X, 0) = 0 \\ C(0, t) = C_0 \end{cases} \quad \begin{matrix} x \in [0, L] \\ C(L, t) = C_0 \end{matrix} \quad (3)$$

Where  $C$  is the concentration of the sulfate *ions*(mol/m<sup>3</sup>),  $x$  is the location of the section( $m$ ),  $t$  is the time of diffusion( $s$ ),

$D_e$  is the Ionic diffusion coefficient of sulfate in concrete ( $m^2/s$ ),  $C_d$  is the dissipation concentration which is caused by the chemical reactions of sulfate *ions*(mol/m<sup>3</sup>),  $C_0$  is the boundary concentration of sulfate *ions*(mol/m<sup>3</sup>),  $L$  is the thickness of the concrete member, and  $[0, L]$  interval represents the concrete cross section [15].

### 1) Chemical Reactions Rate of Sulfate Dissipation:

Reaction-diffusion phenomena is a composition of chemical reactions and ionic diffusivity at the same time which in order to solve the equation, each one of these terms must be evaluate and calculate individually. Reaction term of the Eq.2 can be expressed as:

$$\frac{\partial C}{\partial t} = -k_v \cdot C_{ca^{2+}} \cdot C \quad (4)$$

Where  $K_v$  is the rate of chemical reaction in Eq.4;  $C_{ca^{2+}}$  is the calcium ion concentration in pore solution, and the value of  $k_v \cdot C_{ca^{2+}}$  is assumed to be the rate of reaction ( $k$ ). This can be assumed as a linear function of temperature and is taken as 25mol/m<sup>3</sup> a 273K and 10mol/m<sup>3</sup> at 373K [20].

2) *Diffusion Coefficie*: There are 3 main methods in calculation of diffusion coefficient, considering diffusion coefficient as a:

- Constant value, as Zuo represented his numerical method.
- A time variable value which is explained in Suns paper [21].
- A time-depth variable value which is the most accurate method [22].

Since there is not much difference between the accuracy of these methods and the first method has an acceptable estimation of sulfate ions diffusion, the so first method is utilized to find an analytical solution. By the above assumption the general equation will reform as follows:

$$\frac{\partial C}{\partial t} = D_e \frac{\partial^2 C}{\partial x^2} - kC \quad (5)$$

Where  $C$  is a concentration of the substance,  $kC$  is the rate of removal of diffusing substance and  $k$  is the rate of reaction which is a constant value.

### B. Analytical Solution

In order to solve Eq.5 Danckwerts(1951) has tried a specific method which by the use of a simple transformation reform the reaction-diffusion equation to a case that there would be diffusion without reaction. He has presented his method based on to different types of boundary conditions which are *Dirichlet* and *Neumann*. In the case of sulfate attack, the *Neumann* boundary condition has utilized. *Neumann* boundary conditions are represented as follows:

$$C = 0, t = 0, \text{ at all points in the medium}$$

and also

$$C = C_0, t > 0, \text{ at all points on the surface}$$

In the next step by the assumption of no reaction,  $C_1$  will be the solution of the general equation.

$$\frac{\partial C_1}{\partial t} = D \frac{\partial^2 C_1}{\partial x^2} \quad (6)$$

The answer of the following equation will give the actual value of  $C$ :

$$C = \int_0^t C_1 e^{-kt'} dt' + C_1 e^{-kt} \quad (7)$$

Finally with solving the Eq.6 and pasting in Eq.8 and by doing the calculations with the boundary conditions of Eq.3 the final solution will be expressed as follows:

$$C(x, t) = -\frac{4C_0}{\pi} \sum_{n=0}^{\infty} (a_n \cos(\omega_n x) (k \Psi_n (p-1) + p) k) + C_0 \quad (8)$$

Where  $a_n$ ,  $\omega_n$ ,  $\Psi_n$  and  $p$  are represented as follows:

$$a_n = \frac{(-1)^n}{2n+1}, \omega_n = \frac{(2n+1)\pi}{2L}, \Psi_n = \frac{4L^2}{(-D_e(2n+1)^2\pi^2) - 4kL^2}, p = \exp\left(\frac{t}{\Psi_n}\right)$$

### C. Deep lLearning Method

In this section we introduce our presented deep neural network techniques to find a solution for the reaction-diffusion equation, with the use of produced data, we attempted to teach the neural network.

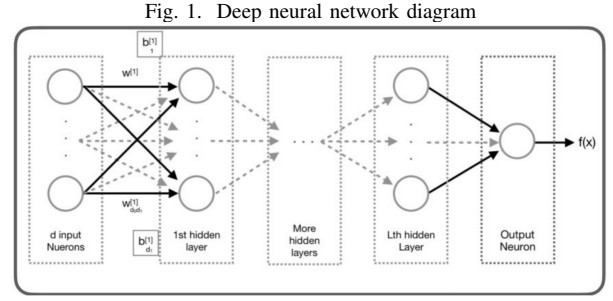
1) *Production data*: To solve the intended reaction-diffusion equation with the use of deep neural networks it is essential to teach the neural network with specific parameters, for the computers to understand the discussed PDE it is needed to extract the critical features of the equation. Since the partial differential equations are defined on the continuous space-time, their domain can be varied from zero to infinity while the computers understand discrete mathematics and limited domains so it is not possible for the computers to understand the issue, Hence it is needed to discretize the period of the features for the computer to understand the function. The features and their period are introduced in the following table:

TABLE II  
DISCRETIZATION INTERVALS AND THE DIMENSION OF T IS IN THE SCALE OF PER YEAR.

$C$	$L$	$X$	$T$	$k$	$D_e$
0 - 200	0 - 0.05	-L - L	0 - 7	$\frac{1}{10} - \frac{1}{10^{10}}$	$\frac{1}{10} - \frac{1}{10^{13}}$

where  $C$  is concentration,  $L$  is half of the domain,  $X$  is the variable position on the domain,  $T$  is the time (year),  $k$  is reaction rate and  $D_e$  is the effective diffusion coefficient.

By the use of *gaussian random* data generation method, 3 million data has produced. After the data production, data were categorized into categories which are called Batch, and 1000 Batches has been produced which each one contains 3000 data. The main idea of categorizing process is to increase the learning model rate. Specific conditions are considered for each parameter to optimize the data distribution so the range of each parameter is discretized into different parts, the data in



each part are chosen randomly with specific steps. The selected steps differ for each part of the input data.

Like the table above, the rest of the parameters based on the modality of each parameter they will be classified and then the data generation process will be done. The input data is made of different parameters which each one of them covers a specific period so, in order to increase the learning rate, the data will be normalized with the following methods:

$$\mu = \frac{1}{m} \sum_{i=0}^n X_i, \sigma^2 = \frac{1}{m} \sum_{i=0}^n X_i^2, x' = x - \mu, x'' = \frac{x'}{\sigma^2} \quad (9)$$

The data generated as  $x$  from Eq.9 will be considered as the input data.

2) *Learning model method*: A 3-layer network is considered for the learning model in which the layers of the network are introduced as follows:

- Input layer.
- Hidden layer.
- Output layer.

The input layer receives the data as a matrix and delivers them to hidden layers. Hidden layers after a process send them to the output layer.

The figure 1 is a general schematic of trained neural network. Where in this network  $w^{[l]}$  is the weight matrix which relates  $l$  and  $l-1$  bit-layers,  $w_{ij}^{[l]}$  is the amount of matrix weight for neuron  $i$  in layer  $l$ .  $j$  is in the layer  $l-1$  and the vector  $b^{[l]}$  is equal to bias amount for layer  $l$ . The equations above demonstrate the updating method of the discussed amounts.

$$Z^{[l]} = W^{[l]} * A^{[l-1]} + B^{[l]} \quad (10)$$

$$A^{[l]} = g^{[l]}(Z^{[l]}) \quad (11)$$

In this kind of situations  $A^{[l]}$  is a matrix which is the amount of the previous layer. The function  $g^{[l]}$  is the utilized activation function in the  $l$ th layer the activation function which used in the hidden layer is function of LeakyReLU and Sigmoid which are explained as follows:

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (12)$$

$$\text{LeakyReLU} = \begin{cases} x & x \geq 0 \\ x * 0.001 & x < 0 \end{cases} \quad (13)$$

First, the values of  $W$  will be randomly chosen between 0 to 1 and then the values of  $B$  will be considered less than zero. The main purpose of learning model method is to decrease the error function.

$$\min_{W,B} J(W, B) \quad (14)$$

$$J(W, B) = \frac{1}{m} \|Y' - Y\|_2^2 \quad (15)$$

Where  $Y'$  and  $Y$  are the estimated value and the exact value which is extracted from the analytical solution. To prevent the *Overfitting* in this part, the error function was changed as follows:

$$J(W, B) = \frac{1}{m} \|Y' - Y\|_2^2 + \frac{\lambda}{2m} \|W\|_2^2 \quad (16)$$

$$J(W, B) = \frac{1}{m} \sum (Y' - Y)^2 + \frac{\lambda}{2m} \sum_{j=1}^{n_x} W_j^2 \quad (17)$$

$$J(W, B) = \frac{1}{m} (Y' - Y)^T (Y' - Y) + \frac{\lambda}{2m} W^T W \quad (18)$$

To increase the learning rate, the Adams optimization method was utilized.

3) *Tuning parameters*: One of the most critical and inevitable parts of the learning algorithm is tuning parameters. In order to achieve a better learning model, it is essential to have an acceptable estimation in the following parameters: 1-  $\alpha$  (learning rate) 2-the number of hidden layers 3-the number of neurons in the hidden layer.

In the next step, its needed to divide the generated data into 3 categories: **test data, validation data, training data**. 90 percent of the 1000 batches were allocated to the numerical solution and considered for the train, and also 5 percent of generated data was used for production validation results by the numerical method in order to fix deep learning parameters. The 5 percents of remaining data were used by the analytical solution to produce test results.

### III. RESULTS

In this section, the accuracy of the deep learning solution has analyzed in comparison to the analytical solution which by changing the critical features of the corresponding equation this could be examined. there are 2 main strategy to examine and evaluate the solutions in the deep learning method:

#### A. Threshold Concept

Firstly the precision of deep learning method and its error with the analytical solutions of the given PDE should be evaluated then the errors must be checked in different learning data, in order to this 1000 batches has been generated which each one contains 3000 data. In each level of examination, the  $X$  number of batches should be selected out of the 1000 given batches. The experiment under the following conditions has been taken: 90% of the given data was considered as training data, 5% as validation data and 5% as test data.

The *Mean Square Error* index has been utilized to calculate the 3 following errors: training error, validation error, and test error. The *MSE* index is represented in Eq.19:

$$MSE = \frac{\sum_{i=0}^n (y' - y)^2}{n} \quad (19)$$

The *Threshold concept* has utilized in order to compare the exact amount of the given PDE with the results form deep learning method. Whereas  $y'$  is the deep learning calculated quantity and  $y$  represents the amount of PDE.

$$|y - y'| < \theta \quad (20)$$

If the threshold quantity was more than left-hand side of Eq.20, then both values will be assumed as equal.

TABLE III  
BATCH NUMBER SENSITIVITY ANALYZE BATCH.

#Batch	Training	Validation	Test	Thr(2)	Thr(1)
100	0.8954	6.1547	9.1574	71.52%	65.14%
300	0.8521	5.9856	7.1259	74.32%	70.78%
500	0.7485	5.0198	5.1245	78.91%	73.15%
700	0.6574	3.1497	4.1547	85.47%	80.19%
1000	0.5782	1.8643	3.0214	91.71%	89.18%

Looking at table.III in more details, by increasing the number of batches training and test errors have been decreased. it is also noticeable that the validation data fall by increasing the number of batches. during the learning process the main aim is to decrease the validation data error which have been satisfied properly.moreover, increment of the batches directly influence the accuracy of the deep learning process.

#### B. Changing Parameter Analyze

In this section, the dependency of the deep neural network to the value of the equations coefficients have been analyzed. In order to conduct this purpose, all of PDEs coefficients were considered as constant values except one of them, and by changing that coefficient, the accuracy of the deep learning solution has been compared to the analytical solution with using three different thresholds.

The variable parameters in this part are  $k$  and  $D_e$  which play a central role in the behavior of the equation. Table.IV demonstrate the accuracy of deep learning solution by changing the  $k$  and  $D_e$  values, where  $C_0$  and  $L$  are 75.5 and 0.05 respectively.

According to the table.IV, it is understood that the accuracy of the solution strongly depends on the correlation of *Rate of reaction* and *Diffusion coefficient*. In the case of constant *Diffusion coefficient* where  $k$  value is less than the particular amount the accuracy of the solution drop considerably. It is also similar for the case of the constant *Rate of reaction* where if  $D_e$  value chose larger than specific value, the solution will not be reliable. When it comes to physics of the phenomena, reduction of the accuracy of the deep learning solution become

TABLE IV  
ACCURACY ANALYZE OF DEEP LEARNING SOLUTION BASED ON  
CHANGING COEFFICIE.

$k$	$Thr(2)$	$Thr(1)$	$Thr(0.5)$	
$2.125 * 10^{-2}$	87.56%	82.39%	79.45%	$D_e = 2.6 \times 10^{-9}$
$2.125 * 10^{-5}$	87.84%	83.05%	80.25%	
$2.125 * 10^{-7}$	88.69%	84.58%	82.12%	
$2.125 * 10^{-10}$	80.32%	78.73%	77.45%	
$2.125 * 10^{-13}$	64.78%	63.58%	61.03%	

$D_e$	$Thr(2)$	$Thr(1)$	$Thr(0.5)$	
$2.6 * 10^{-5}$	73.38%	66.39%	61.45%	$k = 2.125 \times 10^{-7}$
$2.6 * 10^{-7}$	84.29%	82.48%	76.21%	
$2.6 * 10^{-10}$	89.29%	87.78%	82.27%	
$2.6 * 10^{-12}$	88.57%	86.54%	82.12%	
$2.6 * 10^{-15}$	81.33%	75.91%	70.67%	

reasonable. This is because our network was trained for Reaction-Diffusion equation but in these intervals, the equation is changed to the pure reaction and pure Diffusion. To tackle this issue using Dimensional Analysis methods would be useful.

### C. Dimensional Analyze

In order to conduct the pure reaction and Diffusion equation it was needed to change the Eq.(2) to a dimensionless form which is expressed as follows:

$$\frac{\partial C^*}{\partial t^*} = \frac{D_c t_c}{L^2} * \frac{\partial^2 C^2}{\partial x^{*2}} - k t_c C^* \quad (21)$$

where dimensionless parameters are defined as follows :

$$x^* = \frac{x}{L}, t^* = \frac{t}{t_c}, C^* = \frac{C}{C_0}$$

The reason behind using the dimensional analyze is that we want to find out the solution of the pure Reaction and Diffusion equation with a deep learning algorithm which is designed for Reaction-Diffusion Equation. For this purpose, a famous dimensionless number was used for finding suitable coefficients and then, the solution compared to the analytical solution of the pure Reaction equation as a case study.

1) *Damkohler number*: In reaction-diffusion phenomena there is an important dimensionless parameter which is called *Damkohler number* that is defined as follows:

$$D_a = \frac{\text{Rate of reaction}}{\text{Diffusion rate}} \quad (22)$$

For Eq.(21) which is the dimensionless form of the reaction-diffusion equation, Damkohler number is defined as:

$$D_a = \frac{kL^2}{D_c} \quad (23)$$

This number represents the states of reaction-diffusion in different states where  $D_a \cong 1$ ,  $D_a \gg 1$ , and  $D_a \ll 1$  mean the physics of Reaction-Diffusion, pure Reaction, and pure Diffusion respectively.

In the following table, by using the Damkohler number variations, the results of deep learning method with analytical solutions of pure reaction equation has been compared and the errors have calculated:

TABLE V  
DAMKOHLEK VARIATION FOR REACTION EQUATION.

$D_e$	$MSE$	$Thr(0.5)$	$Thr(1)$	$Thr(2)$
$2 * 10^{-14}$	0.256	94.23%	96.95%	98.45%
$2 * 10^{-13}$	1.731	91.14%	95.92%	98.03%
$2 * 10^{-12}$	5.065	90.86%	95.53%	97.75%
$2 * 10^{-11}$	9.077	90.40%	94.93%	96.21%
$2 * 10^{-10}$	11.623	85.32%	86.93%	88.36%

$C_0 = 75.5(mol/m^3)$ ,  $L = 0.05(m)$ ,  $k = 2 \times 10^{-4}(1/s)$

It is needed to note that based on the Damkohler concept, the values for Damkohler must be chosen carefully between specific intervals. In the table.V it is clear that the accuracy of deep learning solution dramatically decreases when the Damkohler number gets smaller from a specific value.

## IV. CONCLUSION

An analytical solution for the Reaction-Diffusion equation was conducted with assumption for the case of the sulfate attack to the concrete, and by using the numerical solution of the equation which is simply available, a deep neural network was trained to predict the behavior of the equation. The pattern recognition feature of the deep learning successfully predict the solution. It is also found that by taking the advantage of dimensional analyze it is possible to estimate the pure Reaction and Diffusion equations with good accuracy.

## REFERENCES

- [1] K. W. Morton and D. F. Mayers, *Numerical solution of partial differential equations: an introduction*. Cambridge university press, 2005.
- [2] B. Zakeri, A. K. Monsefi, S. Samsam, and B. K. Monsefi, "Weakly supervised learning technique for solving partial differential equations; case study of 1-d reaction-diffusion equation;" in *International Congress on High-Performance Computing and Big Data Analysis*. Springer, 2019, pp. 367–377.
- [3] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.
- [4] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.
- [5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [6] G. Hinton, L. Deng, D. Yu, G. Dahl, A.-r. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, B. Kingsbury *et al.*, "Deep neural networks for acoustic modeling in speech recognition," *IEEE Signal processing magazine*, vol. 29, 2012.
- [7] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, p. 484, 2016.
- [8] Y. Bai, L. Guo, L. Jin, and Q. Huang, "A novel feature extraction method using pyramid histogram of orientation gradients for smile recognition," in *2009 16th IEEE International Conference on Image Processing (ICIP)*. IEEE, 2009, pp. 3305–3308.

- [9] P. Liu, J. Gan, and R. K. Chakrabarty, "Variational autoencoding the lagrangian trajectories of particles in a combustion system," *arXiv preprint arXiv:1811.11896*, 2018.
- [10] R. Sharma, A. B. Farimani, J. Gomes, P. Eastman, and V. Pande, "Weakly-supervised deep learning of heat transport via physics informed loss," *arXiv preprint arXiv:1807.11374*, 2018.
- [11] J. Sirignano and K. Spiliopoulos, "Dgm: A deep learning algorithm for solving partial differential equations," *Journal of Computational Physics*, vol. 375, pp. 1339–1364, 2018.
- [12] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, 2017.
- [13] J. Han, A. Jentzen, and E. Weinan, "Solving high-dimensional partial differential equations using deep learning," *Proceedings of the National Academy of Sciences*, vol. 115, no. 34, pp. 8505–8510, 2018.
- [14] A. K. Monsefi, B. Zakeri, S. Samsam, and M. Khashehchi, "Performing software test oracle based on deep neural network with fuzzy inference system," in *International Congress on High-Performance Computing and Big Data Analysis*. Springer, 2019, pp. 406–417.
- [15] X.-B. Zuo, W. Sun, and C. Yu, "Numerical investigation on expansive volume strain in concrete subjected to sulfate attack," *Construction and Building Materials*, vol. 36, pp. 404–410, 2012.
- [16] N. Cefis and C. Comi, "Chemo-mechanical modelling of the external sulfate attack in concrete," *Cement and Concrete Research*, vol. 93, pp. 57–70, 2017.
- [17] C. Sun, J. Chen, J. Zhu, M. Zhang, and J. Ye, "A new diffusion model of sulfate ions in concrete," *Construction and Building Materials*, vol. 39, pp. 39–45, 2013.
- [18] J. Crank *et al.*, *The mathematics of diffusion*. Oxford university press, 1979.
- [19] H. K. Versteeg and W. Malalasekera, *An introduction to computational fluid dynamics: the finite volume method*. Pearson education, 2007.
- [20] J. H. Perry, "Chemical engineers' handbook," 1950.
- [21] Y.-M. Sun, M.-T. Liang, and T.-P. Chang, "Time/depth dependent diffusion and chemical reaction model of chloride transportation in concrete," *Applied Mathematical Modelling*, vol. 36, no. 3, pp. 1114–1122, 2012.
- [22] J. Zuquan, S. Wei, Z. Yunsheng, J. Jinyang, and L. Jianzhong, "Interaction between sulfate and chloride solution attack of concretes with and without fly ash," *Cement and Concrete Research*, vol. 37, no. 8, pp. 1223–1232, 2007.