A Neural Network Approach to Predict Oil Volume Production Considering Porous Media Images

Pedro Calderano
Helon Ayala
Marcio Carvalho
Department of Mechanical Engineering
Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro - RJ, Brazil
pedro.calderano@lmmp.mec.puc-rio.br
helon@puc-rio.br
msc@puc-rio.br

***Abstract*—Two-phase flow through porous media is a complex problem since changes in the porous media geometry can cause a significant impact on the fluid flow. The geometry change affects the system permeability of each phase, which is the coefficient linking the flow rate response to an imposed pressure differential. The phase permeability is also a function of the system saturation (ratio between the phases). This flow is costly to predict through direct simulation methods. Moreover, it is tricky to compute the two-phase flow behavior by simpler methods due to the high variability associated with geometry and fluid properties. In this work, we take 2D porous media images as input. We consider the porous media to be initially fully saturated with oil. Water is injected on a porous media side to displace the oil to the opposite extreme. We assembled a neural network system using the DeepONet concept to predict the volume of oil produced over time. The networks demonstrate to calculate the total oil produced presenting a small relative error.**

***Index Terms*—Neural Networks, VGG-16, DeepONet, Oil Production, two-phase flow, porous media**

I. Introduction

Porous media are solid structures with empty regions, which can be connected, such as rocks and sponges. In the cases where the voids are connected, fluid flow can happen, which produces applications ranging from filtration to aquifer exploration to petroleum extraction. Oil recovery is a process that submits the oil stored in rocks to a pressure differential to force the fluid to come out through the production well. Therefore, this activity associated with porous structures leads to the investigation of flow through rock-like porous media.

The inference time of machine learning methods is significantly lower than the convergence time of direct numerical methods that solve physical modeling. Aiming to save computational resources, several works have been employing machine learning algorithms to approximate certain functions, parameters, or system states. Recently, some works employing computational intelligence appeared to infer rock properties related to fluid flow. Wu et al. [1] used convolutional neural networks (CNN) to estimate the permeability, which can be interpreted as the inverse of the resistance to fluid flow, of a 2D porous medium image. The traditional approach would require a flow simulation using a direct solver. As permeability is the measure linking the flow rate to the pressure differential, the simulation solution allows us to calculate the permeability associated with the image. [1] took the CNNs characteristic of learning patterns from images and built a CNN architecture to infer the permeability associated with the porous media image. The porosity and the surface area of porous media were also accounted as inputs of the [1] networks. Wang et al. [2] developed a U-Net model [3] to assist the direct simulation of fluid flow in porous media. Although the fluid flow prediction using the network is not accurate enough, it produces a field that is a good initial guess for the direct simulation method, reducing its convergence time. Tembely et al. [4] applied machine learning methods to predict rock permeability based on 3D images obtained using microtomography. Zhou et al. [5] employed a super-resolution network to improve the flow field of a coarse grid direct simulation. In this way, [5] performed a cheap direct simulation to be improved by the neural network system. Santos et al. [6] developed a deep neural network to predict the flow velocity field in 3D porous media images.

Although some works regarding flow through porous media have been developed, we have not found any work concerning the flow properties estimation in two-phase flow systems using machine learning tools directly from porous media images. The physics behind two-phase flow in porous media involves capillarity effects [7] that appear on the interface of fluids and viscous effects. As capillary forces depend on interface curvature, flow streams of a displacing fluid meeting multiple channels may not displace the initial fluid of all channels. As pressure builds up and the displacing fluid invades the neighboring channels, the resisting channel cannot have its filling fluid displaced since there will not be a pressure differential between its extremities. This behavior brings high variability in the two-phase fluid flow in porous media and makes system analysis complex. Water flooding is a common technique of enhanced oil recovery [8], [9]. This

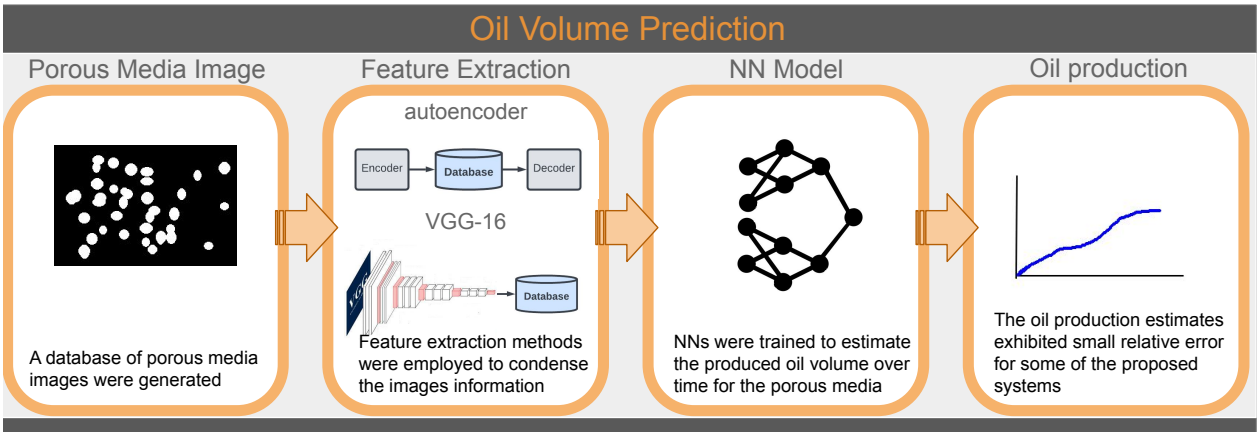

Fig. 1. Graphical overview of the developed system. We generate a dataset of porous media to estimate the volume of oil displaced by a water injection process.

technique consists of injecting water into a well to improve the oil production at the producing oil well. The two-phase flow in a porous media of water displacing the oil stored in the rock happens in the pore-scale of the water flooding process. The determination of the oil recovery factor, which determines the percentage of the oil that is stored in the porous media can be extracted. This parameter provides relevant information about a reservoir regarding its management. This factor provides a reference to the volume of resources that can be exploited in a field. There are experimental and numerical approaches to performing the estimation of the oil recovery factor [10]–[14]. The experimental process involves displacing oil in a porous medium with another fluid, which is expensive and time-consuming [10]. The numerical approach uses inferences from statistical methods [10]–[13], which rely on measurement data to develop a model. Han et al. [10] proposed a model based on support vector machines combined with particle swarm optimization. Their analysis evaluated geological variables, such as permeability, to predict the oil recovery factor. Other works [15]–[19] developed models analyzing the variables and their importance to their prediction.

CNNs are notoriously known for extracting patterns from images [20]–[24] and performing tasks, such as image classification, semantic segmentation, and object detection. Therefore, it is natural to try to use these architectures to develop systems that consider image-like inputs. It is possible to train entire networks from scratch or leverage pre-trained networks e.g. [25], [26], by fine-tuning or freezing their weights and possibly adding extra layers tailored to the desired application. Another network architecture is the DeepONet [27], that was recently introduced to approximate an operator between two independent functions. Therefore, this architecture can build a structured connection between the representation of different measures connected by the problem, such as the domain variables and their boundary conditions.

The goal of this paper is to introduce a machine learning-based method to predict the volume of oil displaced by the injection of another fluid into the porous medium. This machine learning algorithm will predict the recovered volume only from tomography-like images. We propose convolutional networks and a DeepONet structure to perform the desired intent consuming lower computational resources than the traditional simulation methods. The proposed methods presented relatively good performance considering the metrics adopted.

The remainder of this work is organized as follows. Section II refers to the dataset used to develop the networks. Section III discusses the methods applied, and Sec. IV presents the discussion on the results. Finally, Sec. V states the conclusions.

II. Dataset Assembly

The dataset employed in this work was constructed using the results from Finite Element Method direct simulations. It considers porous media initially filled with oil that is displaced by the injection of another fluid. To build the dataset, first, we generated images representing some 2D porous media. These images were generated by setting a number of ellipses in random positions. The image represents a porous media with 14 mm height and 21 mm width as dimensions in 165 by 248 pixels. The number of ellipses and their sizes ranges from 30 to 70 and 0.8 to 1.5 mm, respectively. We used uniform random distributions to define these values. The conceived images have a black background with white shapes distributed through its extension, as shown in Figure 2. The white forms in the Figure 2 represent solid grains, where fluid cannot flow through, while the black region represents empty spaces or voids, where the fluid flow happens. Then, we build the meshes based on the images. On the meshes, we assign inlet boundary conditions on the image's left side, no-slip conditions on the top and bottom sides and the edges of the solid grains, and a pressure point on a point

in the bottom right corner. The inlet boundary condition is a uniform pressure imposed. The no-slip condition imposes null velocity. The pressure point establishes the referential point where pressure is zero. Therefore, the inlet pressure (p_{in}) causes a pressure differential relative to the referential point (p_o). This pressure differential ($dP = p_{in} - p_o$) drives the flow. The inlet boundary condition considers the saturation of the displacing fluid as 1, which means that the displacing fluid is injected with no mixture (single phase).

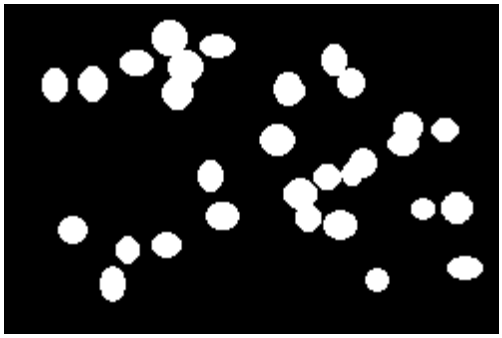

Fig. 2. Sample of image generated to simulate a porous media. The white pixels represent solid grains, while the black pixels represent empty spaces or voids.

The simulation using the Finite Element Method considered the momentum conservation and continuity equations present in the Navier-Stokes formulation to solve the flow field, Eqs. 3, 4, and 5. The interaction between the two phases is computed by the chemical species diffusion equation, Eq. 6. In the equation system presented in Eqs. 1-6, the term ρ is relative to the fluid density, μ to the fluid viscosity, t to time coordinate, x and y to spacial coordinates, u and v the velocity in the x and y directions, p to the pressure, and Sw is the injected fluid saturation. The indexes 1 and 2 represent the injected fluid phase and the initial oil phase respectively.

$$\rho = S_w\rho_1 + (1 - Sw)\rho_2 \tag{1}$$

$$\mu = S_w\mu_1 + (1 - Sw)\mu_2 \tag{2}$$

$$\rho\left(\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) = -\frac{\partial p}{\partial x} + \mu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \tag{3}$$

$$\rho\left(\frac{\partial v}{\partial t} + v\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right) = -\frac{\partial p}{\partial y} + \mu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) \tag{4}$$

$$0 = \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \tag{5}$$

$$\frac{\partial Sw}{\partial t} + \mathbf{v}\cdot\nabla S_w = D\nabla^2 S_w \tag{6}$$

We considered the physical parameters in the simulation as $\rho_1 = \rho_2 = 1000\,[kg/m^3]$, $\mu_1 = 0.01\,[Pa.s]$, $\mu_2 = 0.1\,[Pa.s]$, $D = 1E-10\,[m^2/s]$, $dP = 1\,[Pa]$. The simulations were run for 200 ($\Delta t = 2s$) time steps. Figure 3 shows the saturation at the last time step of the simulation in the porous medium presented in Figure 2. The green area is filled with the injected fluid, while the blue area is where there is the reminiscent oil in the porous medium.

The constructed dataset is composed of 2274 samples. The porous media images are the raw input to the system to be developed, while the output is the volume of oil produced over time. The produced oil volume is computed using Eq. 7.

$$V(t = T) = \int_0^T \int_0^h S_w(t, x, y)u(t, x, y)dydt|x = 0.021 \tag{7}$$

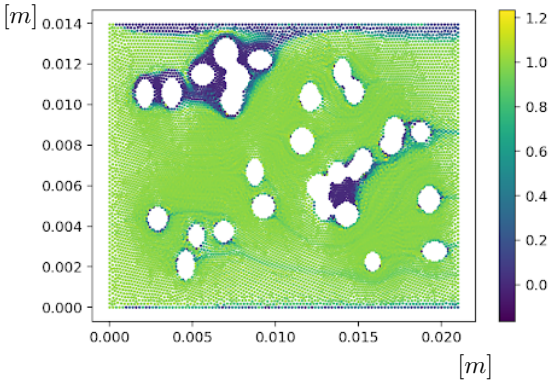

Fig. 3. Last simulation time step saturation. The blue region corresponds to places where the oil is trapped, while the green region corresponds to the area filled with the injected fluid. In the initial time step, the porous medium is fully saturated with oil (the image would be entirely blue for $t = 0$). The fluid flows from left to right.

III. Methodology

In this work, we use feature extraction techniques as a pre-processing step. The networks trained by pre-processed features consume less memory during their training phase and have a greater propensity to converge quicker than networks trained directly by the porous media images. We used two pre-processing tools to the image dataset, a trained autoencoder, and the pre-trained VGG-16 [25] network, which will be explained in Subsec. III-A and III-B. The pre-processing techniques produce separated processed datasets. Subsec. III-C and III-D presents the networks employed to predict the produced oil volume signal, which are a convolutional network and a network based on the DeepONet concept.

A. Autoencoder

Autoencoders [28] are neural networks designed to take some information on their input and give back similar information in their output. Its structure is composed of an encoder that condenses the input features and then a decoder that expands the information back. The encoder part of the

network takes the input and has a descending number of neurons in each of the encoder layers until it reaches the bottleneck, which is the layer with the least number of neurons in the entire network. The output of the encoder can be interpreted as a representation of the input in a space with reduced dimensions. Therefore, the encoder's output can be used to transform a dataset's features. The decoder part of the network has an ascending number of neurons in each layer in a way that the entire network is symmetrical in relation to an axis placed in the bottleneck layer. The output of the decoder should be a representation of the input in the same space. So, this network concept can also be employed for denoising purposes. In this work, we use the autoencoder concept to have a representation of the input images in a reduced feature space.

The first layer of the constructed autoencoder flattens the 165 by 248 pixels of the image matrix in a vector and has the same number of neurons as the number of pixels. The second layer is the bottleneck with 1000 neurons. Finally, the third and last layer has the same number of neurons as the first layer and transforms their output back into the 165 by 248 matrix. So, this autoencoder transforms the images to a 1000 dimensions space. Then, the autoencoder transforms the dataset into a 1000 features dataset. This autoencoder was trained using the binary cross-entropy loss, using the Adam optimizer with learning rate $lr = 1E-3$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. We take 30% of the data to perform a validation process during the training phase. The validation considered the mean squared error as its metric. The autoencoder was trained for 500 epochs without early stopping criteria. Figure 4 and Figure 5 show the loss and the validation values across the training epochs.

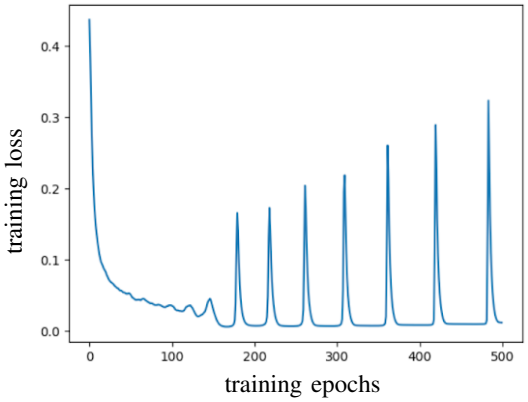

Fig. 4. Binary cross entropy loss evolution over training epochs for the autoencoder training phase.

B. VGG-16

The VGG-16 [25] is a network introduced to classify images of the imagenet [29] challenge. The VGG-16 architecture is split into two parts. Its first part is composed of convolutional layers, while fully connected layers compose its last part. We take the output of the convolutional layers to transform the image dataset from a 165 by 248 matrix to a 5 by 7 by 512 matrix. This transformation not only compresses the dataset features but also puts the dataset in a latent space that has been successful as a feature extraction tool for different problems [30]–[32]. Therefore, we employ the use of this tool as a pre-transformation of the input dataset. The imagenet VGG-16 was employed only as a feature extraction method, meaning that we do not perform fine-tuning to its weights.

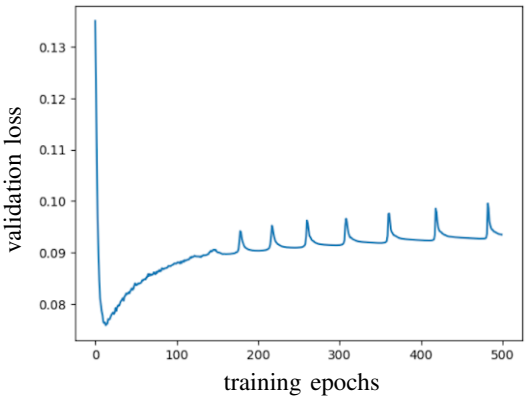

Fig. 5. Mean squared error loss evolution relative to the validation group over training epochs for the autoencoder training phase.

C. Convolutional Network (CNN)

Convolutional Networks (CNN) [33], [34] are the standard approach to dealing with image-like data. Therefore, we use a convolutional network approach to estimate the oil volume production over time. In this approach, we take the dataset formed through the VGG-16 network. This dataset passes through convolutional layers with filter size (2, 2) with hyperbolic tangent activation function and padding. The final layer has filter size (5, 7) with either ReLU or no activation function and does not consider padding. All convolutional layers have 200 filters. Table I compiles the configurations of activation functions tested for the convolutional networks.

activation functions by layer
$[tanh\ none]$
$[tanh\ relu]$
$[tanh\ tanh\ none]$
$[tanh\ tanh\ relu]$

TABLE I

THE TABLE RESUMES THE CONVOLUTIONAL NETWORKS TESTED. IT SHOWS THEIR LAYER'S ACTIVATION FUNCTIONS. EACH LAYER HAS 200 FILTERS.

D. DeepONet Network

The Deep Operator Network (DeepONet) [27] is a configuration of the network developed to approximate operators G that maps two independent inputs u and y, $G(u)(y)$. [27] shows using theorems that $G(u)(y)$ can be approximated through the dot product between functions of those inputs, $G(u)(y) \approx f(u) \cdot g(y)$. Then, [27] extrapolates the theorem to

neural networks. Two separate neural networks approximate the $f(u)$ and $g(y)$ functions. The vector output of those networks is merged through a dot product operation to reproduce the theorem statement. The functions $f(u)$ and $g(y)$, as well as the networks that approximate them, are called branch and trunk functions.

Inspired by the DeepONet concept, we developed a network that takes as input information from the porous media geometry in the branch network and trunk network the values considering the previous oil volume predictions with the time step of the current prediction. Figure 6 exhibits the structure of the DeepONet network. As shown in the scheme, this network takes three inputs to perform a prediction. The 'input_img' is the input of the branch network. This input can be either the dataset transformed by autoencoder or the dataset transformed by the VGG-16. The trunk network takes two inputs. Let us name the predicted signal as $\mathbf{q} = \big[q(1), q(2), \cdots, q(k-1), q(k), q(k+1), \cdots, q(200)\big]$. The 'input_previous' in the trunk network takes the previous 15 time steps $\mathbf{q}$ values to predict the volume in the next time step. Then, we take $\big[q(k-14), q(k-13), \cdots, q(k-1), q(k)\big]$ to predict $q(k+1)$. Therefore, the prediction in this network type is interactive, since it needs to know information about previous time steps to infer the next one. The 'input_time' is also in the trunk network and has the $k+1$ value. We evaluate the network performance by changing the number of neurons ('n_neurons') in the Dense layers, and the number of filters ('n_filters') in the convolutional layer. The convolutional layer considers padding and the ReLU activation function. The Dense layers consider the hyperbolic tangent activation function, except the last two Dense layers merged through the dot product, which adopts the ReLU activation function.

IV. Results

This work adopts the methods described in Sec. III to deal with data from the porous media images and estimate their oil volume production. We first transform the dataset using the autoencoder, described in Subsec. III-A, and the VGG-16, described in Subsec. III-B. Then, we use the two transformed input datasets in the CNN and DeepONet systems, described in Subsecs. III-C and III-D.

The dataset used to train and evaluate the system is composed of 2274 cases, as described in Sec. II. This dataset was split into three to perform the training, validation, and testing tasks. Those datasets are composed of 1455, 364, and 455 cases, respectively. All systems were trained for a maximum of 250 epochs, with stopping criteria of 50 validation checks without a loss improvement. The loss considered for both the training and validation processes was the Mean Squared Error (MSE). The training procedure employs 256 samples in each batch. We fix the algorithm's random seeds to ensure the results reproducibility. The CNN training process adopted the Adam optimizer with learning rate $lr = 1E-4$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. Meanwhile, it was employed the Adam optimizer with learning rate $lr = 9E-4$, $\beta_1 = 0.9$, and $\beta_2 = 0.999$ to train the DeepONet.

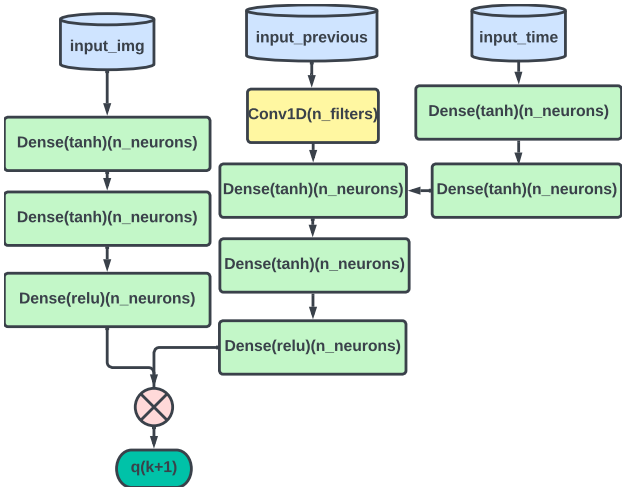

Fig. 6. Sketch of the DeepONet assembled. This network takes in three inputs, respectively 'input_img' with information of the porous medium geometry, 'input_previous' with the previous oil volume predictions information, and 'input_time' with time step information. According to the DeepONet concept, the network line emerging from 'input_img' is the branch network, while the line related to 'input_previous' and 'input_time' is the trunk network.

The model evaluation considers the MSE and the R-Squared coefficient (R2) of the signal predicted. We consider the total volume of oil produced error as metric, given by Eq. 8, in which N_t is the number of samples used for testing purpouses, V_o is the volume of oil produced at the last time step, and $\overline{V_o}$ is the predicted volume of oil produced at the last time step.

$$\epsilon_{oil} = \frac{\sum_{i=0}^{N_t}(V_o - \overline{V_o})_i / V_{oi}}{N_t} \tag{8}$$

Table II displays the metrics for the CNN systems developed, while Tab. III shows the metrics for the DeepONet systems developed. The CNN hyperparameters adopted are described in Subsec. III-C. We vary the DeepONet number of neurons in the Dense layers from 10 to 90 neurons in each layer. The number of filters in the convolutional layer is fixed at 20 filters. Considering the three metrics analyzed, the best model is the DeepONet with 70 neurons in each layer and its input transformed by the VGG-16. Figures 7 and 8 show the volume over time estimate for the best CNN model and the best DeepONet. The DeepONet model not only presented the best metric values but also shows a smooth prediction, in contrast with the CNN model prediction, which is considerably noisy.

V. Conclusion

This work shows an approach to predict the oil volume displaced by fluid injection in porous media using Neural Networks. The dataset employed to develop the system was

CNN system metrics			
	MSE	R2	ϵ_{oil}
[*tanh none*]	0.002	0.866	0.073
[*tanh relu*]	0.091	-1852.353	1.000
[**tanh tanh none**]	**0.002**	**0.887**	**0.070**
[*tanh tanh relu*]	0.055	0.608	0.070

TABLE II
METRICS EVALUATED TO THE CNN SYSTEM. THE INPUT FOR THIS SYSTEM IS DATASET TRANFORMED BY THE VGG-16.

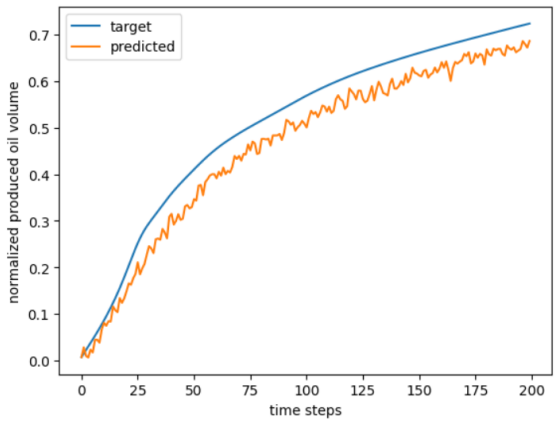

Fig. 7. Prediction of the produced oil volume performed by the CNN model for a test sample.

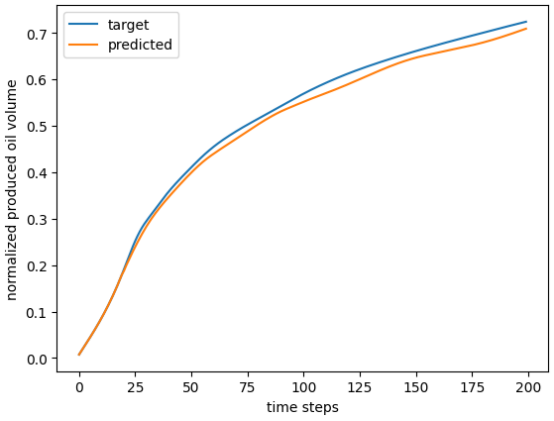

Fig. 8. Prediction of the produced oil volume performed by the DeepONet model for a test sample.

constructed using the Finite Element Method to solve equations of mass and momentum conservation, and the diffusion equation. We propose a system that takes images representing 2D porous media tomography. These images are subjected to a feature extraction method that transforms the image to a space that makes it easier to train the Neural Networks. Then, we use the transformed dataset in the proposed Networks. The proposal consists of a convolutional network and a DeepONet to predict the volume of oil produced at the end of the porous media. The combination of hyperparameters tested achieved an apparent better performance in the DeepONet model. Moreover, the prediction of the DeepONet showed to be smoother than the convolutional system prediction.

Future works should incorporate an initial prediction method in the DeepONet system since it needs previous steps information. Then, it does not predict the first time steps. We also plan to perform statistical measures of the models to offer a more solid base to compare the achieved results. Moreover, we should explore a dataset with porous media of lesser permeability and porosity values, as explored by [10].

ACKNOWLEDGEMENTS

The authors would like to thank the financial support from the National Agency of Petroleum, Natural Gas and Biofuels – ANP, from Petrobras, and from the Brazilian National Council for Scientific and Technological Development – CNPq.

REFERENCES

[1] J. Wu, X. Yin, and H. Xiao, "Seeing permeability from images: fast prediction with convolutional neural networks," *Science bulletin*, vol. 63, no. 18, pp. 1215–1222, 2018.

[2] Y. D. Wang, T. Chung, R. T. Armstrong, and P. Mostaghimi, "Ml-lbm: predicting and accelerating steady state flow simulation in porous media with convolutional neural networks," *Transport in Porous Media*, vol. 138, no. 1, pp. 49–75, 2021.

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

[4] M. Tembely, A. M. AlSumaiti, and W. Alameri, "A deep learning perspective on predicting permeability in porous media from network modeling to direct simulation," *Computational Geosciences*, vol. 24, pp. 1541–1556, 2020.

[5] X.-H. Zhou, J. E. McClure, C. Chen, and H. Xiao, "Neural network–based pore flow field prediction in porous media using super resolution," *Physical Review Fluids*, vol. 7, no. 7, p. 074302, 2022.

[6] J. E. Santos, D. Xu, H. Jo, C. J. Landry, M. Prodanović, and M. J. Pyrcz, "Poreflow-net: A 3d convolutional neural network to predict fluid flow through porous media," *Advances in Water Resources*, vol. 138, p. 103539, 2020.

[7] M. Blunt, M. J. King, and H. Scher, "Simulation and theory of two-phase flow in porous media," *Physical review A*, vol. 46, no. 12, p. 7680, 1992.

[8] A. Katende and F. Sagala, "A critical review of low salinity water flooding: Mechanism, laboratory and field application," *Journal of Molecular Liquids*, vol. 278, pp. 627–649, 2019.

[9] Y. Chen, Q. Xie, A. Sari, P. V. Brady, and A. Saeedi, "Oil/water/rock wettability: Influencing factors and implications for low salinity water flooding in carbonate reservoirs," *Fuel*, vol. 215, pp. 171–177, 2018.

[10] B. Han and X. Bian, "A hybrid pso-svm-based model for determination of oil recovery factor in the low-permeability reservoir," *Petroleum*, vol. 4, no. 1, pp. 43–49, 2018.

[11] S. Macary and W. A. Al Hamid, "Creation of the fractional flow curve from purely production data," in *SPE Annual Technical Conference and Exhibition*, OnePetro, 1999.

[12] S. Al-Fattah and R. Startzman, "Predicting natural gas production using artificial neural network," in *SPE hydrocarbon economics and evaluation symposium*, OnePetro, 2001.

[13] A. Okpere and C. Njoku, "Application of neural networks in developing an empirical oil recovery factor equation for water drive niger delta reservoirs," in *SPE Nigeria Annual International Conference and Exhibition*, OnePetro, 2014.

[14] L. P. Dake, *The practice of reservoir engineering (revised edition)*. Elsevier, 2001.

[15] I. Makhotin, D. Orlov, D. Koroteev, E. Burnaev, A. Karapetyan, and D. Antonenko, "Machine learning for recovery factor estimation of an oil reservoir: A tool for derisking at a hydrocarbon asset evaluation," *Petroleum*, vol. 8, no. 2, pp. 278–290, 2022.

	DeepONet system metrics					
	autoencoder transformed input			VGG-encoded input		
$\#neu$	MSE	R2	ϵ_{oil}	MSE	R2	ϵ_{oil}
10	0.010	0.725	0.261	0.004	0.854	0.111
20	0.005	0.616	0.129	0.003	0.912	0.162
30	0.008	0.773	0.237	0.002	0.928	0.091
40	0.002	0.857	0.118	0.005	0.477	0.164
50	**0.002**	**0.907**	**0.115**	0.017	0.708	0.342
60	0.006	-0.301	0.220	0.010	0.792	0.092
70	0.008	-0.267	0.237	**0.001**	**0.951**	**0.063**
80	0.002	0.845	0.123	0.026	0.652	0.343
90	0.018	0.500	0.541	0.038	0.528	0.458

TABLE III
METRICS EVALUATED TO THE DEEPONET SYSTEM.

[16] M. Kumar, K. Swaminathan, A. Rusli, and A. Thomas-Hy, "Applying data analytics & machine learning methods for recovery factor prediction and uncertainty modelling," in *SPE Asia Pacific Oil & Gas Conference and Exhibition*, OnePetro, 2022.

[17] T. H. Vo, D. S. Dashtgoli, H. Zhang, and B. Min, "Machine-learning-based prediction of oil recovery factor for experimental co2-foam chemical eor: Implications for carbon utilization projects," *Energy*, p. 127860, 2023.

[18] A. A. Mahmoud, S. Elkatatny, W. Chen, and A. Abdulraheem, "Estimation of oil recovery factor for water drive sandy reservoirs through applications of artificial intelligence," *Energies*, vol. 12, no. 19, p. 3671, 2019.

[19] D. M. Noureldien and A. H. El-Banbi, "Using artificial intelligence in estimating oil recovery factor," in *SPE North Africa Technical Conference and Exhibition*, OnePetro, 2015.

[20] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE transactions on neural networks and learning systems*, 2021.

[21] A. Dhillon and G. K. Verma, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*, vol. 9, no. 2, pp. 85–112, 2020.

[22] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[23] Q. Liu, N. Zhang, W. Yang, S. Wang, Z. Cui, X. Chen, and L. Chen, "A review of image recognition with deep convolutional neural network," in *Intelligent Computing Theories and Application: 13th International Conference, ICIC 2017, Liverpool, UK, August 7-10, 2017, Proceedings, Part I 13*, pp. 69–80, Springer, 2017.

[24] H. Ajmal, S. Rehman, U. Farooq, Q. U. Ain, F. Riaz, and A. Hassan, "Convolutional neural network based image segmentation: a review," *Pattern Recognition and Tracking XXIX*, vol. 10649, pp. 191–203, 2018.

[25] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[27] L. Lu, P. Jin, G. Pang, Z. Zhang, and G. E. Karniadakis, "Learning nonlinear operators via deeponet based on the universal approximation theorem of operators," *Nature machine intelligence*, vol. 3, no. 3, pp. 218–229, 2021.

[28] D. Bank, N. Koenigstein, and R. Giryes, "Autoencoders," *arXiv preprint arXiv:2003.05991*, 2020.

[29] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[30] D. Theckedath and R. Sedamkar, "Detecting affect states using vgg16, resnet50 and se-resnet50 networks," *SN Computer Science*, vol. 1, pp. 1–7, 2020.

[31] A. Krishnaswamy Rangarajan and R. Purushothaman, "Disease classification in eggplant using pre-trained vgg16 and msvm," *Scientific reports*, vol. 10, no. 1, pp. 1–11, 2020.

[32] E. Rezende, G. Ruppert, T. Carvalho, A. Theophilo, F. Ramos, and P. d. Geus, "Malicious software classification using vgg16 deep neural network's bottleneck features," in *Information Technology-New Generations: 15th International Conference on Information Technology*, pp. 51–59, Springer, 2018.

[33] Y. LeCun, Y. Bengio, *et al.*, "Convolutional networks for images, speech, and time series," *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[34] S. J. Prince, "Understanding deep learning," 2023.