InterPore JOURNAL

Original Research

USING GRAPH NEURAL NETWORKS TO PREDICT THE PERMEABILITY OF POROUS MEDIA

Jack M. I'Anson[1,2], Mark J. H. Simmons[1], E. Hugh Stitt[2], Robert W. Gallen[2]

[1]University of Birmingham, School of Chemical Engineering, Birmingham, UK; [2]Johnson Matthey Technology Centre, Billingham, UK

Correspondence to:
Jack M. I'Anson,
JMI179@student.bham.ac.uk

How to Cite:
I'Anson, J., Simmons, M., Stitt, H., & Gallen, R. (2025). Using Graph Neural Networks to Predict the Permeability of Porous Media. *InterPore Journal*, 2(3), IPJ250825-2. https://doi.org/10.69631/ipj.v2i3nr47

RECEIVED: 14 Oct. 2024
ACCEPTED: 18 Jun. 2025
PUBLISHED: 25 Aug. 2025

ABSTRACT

The permeability of porous media is often calculated using correlations or computationally expensive simulations. Several methods have been developed which use neural networks to predict porous media properties, but little work has been done on the development of a model that can handle porous media at the representative elementary volume (REV) scale. This work describes the framework for developing a graph neural network (GNN) to predict the permeability of porous media based on representative pore networks extracted from the structures, rather than representative structure volumes. This allows for consistent input sizes for the neural network, irrespective of the average pore size, which is more difficult when the entire voxelized structure is the model input. A GNN was trained to predict the permeability of porous media based on lattice Boltzmann method (LBM) simulations of the flow through the structures. The GNN showed a good agreement with the LBM simulations over samples with permeabilities spanning several orders of magnitude. The GNN was able to outperform the Carman-Kozeny equation with a mean squared error (MSE) for the unseen testing dataset of 0.00190 and a mean absolute error (MAE) of 0.0302 compared to an MSE of 1.125 and an MAE of 0.783 for the Carman-Kozeny equation, when comparing against the LBM ground truth. The inference time of the GNN alone was several orders of magnitude faster than the LBM simulations, and nearly 10 times faster when including the pore network extraction time needed for the GNN. This work demonstrates the potential of using GNNs to predict the permeability of representative porous media, and the benefits of using model architectures that take pore networks as the inputs.

KEYWORDS

Porous media, Lattice Boltzmann method, Machine learning, Graph neural networks

1. INTRODUCTION

Understanding the properties of porous media is beneficial for several important applications, including fuel cells (62), batteries (84), geology (35), and automotive exhaust treatment (42). Typical methods for numerically determining the properties of porous media, such as effective gas diffusivity or permeability, involve pore-scale fluid simulations. These are split into different approaches, such as traditional computational fluid dynamics (CFD), including the finite volume method (42, 54, 83) and the finite element method (60), and fast Fourier transforms for direct numerical simulation (10, 52). The lattice Boltzmann method (LBM) is an alternative CFD technique that has commonly been used for porous media simulations (20, 76, 79). The LBM can be second order accurate (47) and complex structures can be modeled (44), making it ideal for porous media simulations. However, traditional CFD techniques and the LBM require significant computational cost, often requiring the use of high performance computing (HPC) (20, 83). Pore network models have also been used to determine the properties of porous media (8, 59). Common approaches for constructing pore network models use some type of porosimetry, however, variations in pore size within the structure can lead to inaccurate pore size distributions (81). Alternatively, permeability can be measured experimentally by driving flow through a porous structure, measuring the pressure drop and determining the permeability using Darcy's law (75). While this is useful for validating simulations, it requires a complex experimental setup, and more results can be gathered via simulation.

To overcome the need for extra simulations or experiments, various data-driven methods have been used for predicting porous media properties. A common method is the convolutional neural network (CNN) (3, 68, 76, 79), with both 2D and 3D implementations showing good predictive performance. As with many neural networks, training of the models requires large datasets, and obtaining a sufficiently large dataset from real porous media is difficult due to the time required to image the structures (42). Furthermore, depending on scale, some imaging techniques can be destructive to the samples (62), which is an issue when trying to generate datasets with more than 1000 samples due to waste and cost considerations. Methods of producing large datasets from fewer or no real structures have been developed, including the generation of a dataset by augmenting a pair of real structures (68), extracting smaller, overlapping samples from larger structures (76), or generating a dataset of entirely non-real structures (63, 77, 79, 83). Using methods such as these allows for large datasets with high variance to be generated quickly and cheaply.

A key step in the development of data-driven methods is feature engineering, which provides descriptions of the underlying dataset and improves the degree to which machine learning models can be explainable (18). The features also play a significant role in describing how well the machine learning model can be generalized to new, unseen structures (55). It is therefore important to decide how to best represent the structures in a way that can be interpreted by machine learning methods. As mentioned previously, a common approach is to take 2D (or 3D) images of porous structures and train a CNN. The 2D CNN has shown good results for predicting the properties of 2D images of porous media. However, characteristics of the 2D images are not necessarily present in 3D structures. For example, dead-end pores are often found in 2D images, but are rarely seen in 3D images (36). This means features of 2D structures learned in the machine learning models will not necessarily carry over to the 3D structures. Consequently, the 3D CNN has become more popular for porous media research. Wang et al. (76) trained a 3D CNN to predict the effective diffusivity of randomly generated 3D porous media, showing good predictive performance. However, a general problem with CNNs is that they are computationally expensive, especially in 3D (13), which may prove problematic when trying to predict properties of samples large enough to be representative of the structure they were sampled from.

Although CNNs take in entire structures as inputs, other structural properties may be useful to the machine learning models. For example, a common choice for obtaining information about a porous structure is to extract the pore network (an example pore network is shown in **Fig. 1**). The pore network consists of pore coordinates and pore connections, which can be represented as a graph data structure (29), thereby encoding both pore features and connectivity within a single structure. Graph neural

networks (GNNs) are a class of neural networks specifically designed to process graph data structures (80). They have seen a variety of applications including social recommendation systems (21), journal paper citation trends (16), and text classification (53). However, they have also been used to predict properties of a variety of physical systems, including the quantum properties of molecules (26), and the properties of polycrystalline structures (17). Dai et al. (17) represented a structure of crystals with a graph, storing information about the crystals in a feature matrix and information about the crystal connectivity in an adjacency matrix. The adjacency and feature matrices were then passed to the GNN, making use of message passing layers to determine the influence of neighboring crystals on each other to predict structural properties (24).

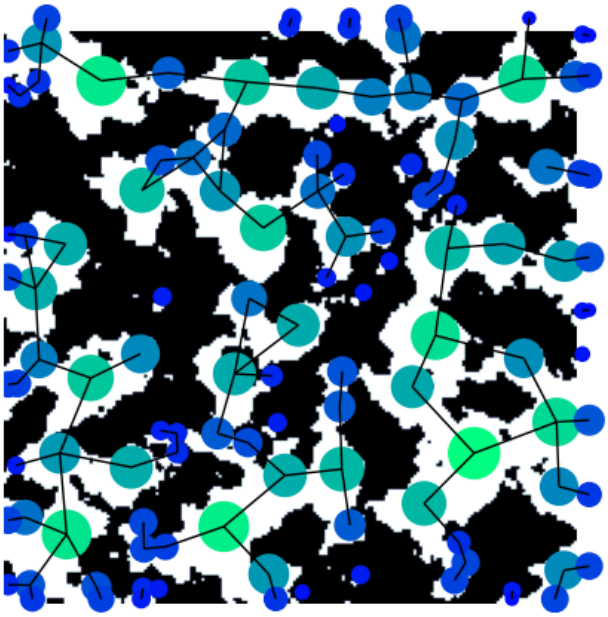

Figure 1: Example 2D pore network generated with PoreSpy (30) and OpenPNM (29). The size and color of the circles represent pores of equivalent size and the lines represent pore connections.

Cai et al. (10) used a graph isomorphism network, a type of GNN, in order to predict formation factor and effective permeability using Morse graphs extracted from porous media images. A Morse graph is extracted from the distance transform, with features such as coordinates and distance values assigned to the graph nodes. This produces a reduced dimension representation of the structure that contains information about each of the pore regions and pore connectivity. It therefore aids in understanding how the neural network obtains information about the structures, leading to a more interpretable machine learning model. Graph neural networks have also been coupled with CNNs to make predictions of porous media properties (4, 86). They aim to capture the strengths of both models: the CNN extracts detailed surface properties from the images, while the GNN captures the connectivity of the structure. However, this often results in highly complex models.

While several methods exist to predict the properties of porous media, it is crucial that they are sufficiently scalable to handle structures large enough to be representative of the original materials from which they were extracted. Training machine learning models on representative 2D samples has been explored (79). However, limited research has been done on the use of 3D samples that are shown to be representative in the CNN literature (70) and, as far as we are aware, no research has been done on the use of representative pore networks for training a GNN. This work addresses the problem by training a GNN on representative pore networks extracted from the samples, enabling it to generalize to previously unseen porous media.

First, a set of large, unique, non-real structures that mimic the properties of real porous media were generated. Lattice Boltzmann method simulations were then performed on the large structures. Following this, smaller, representative samples were extracted from the larger structures and the simulation results, creating a large and varied dataset. Finally, a GNN was trained to predict the permeability of the samples based on the pore network extracted from within.

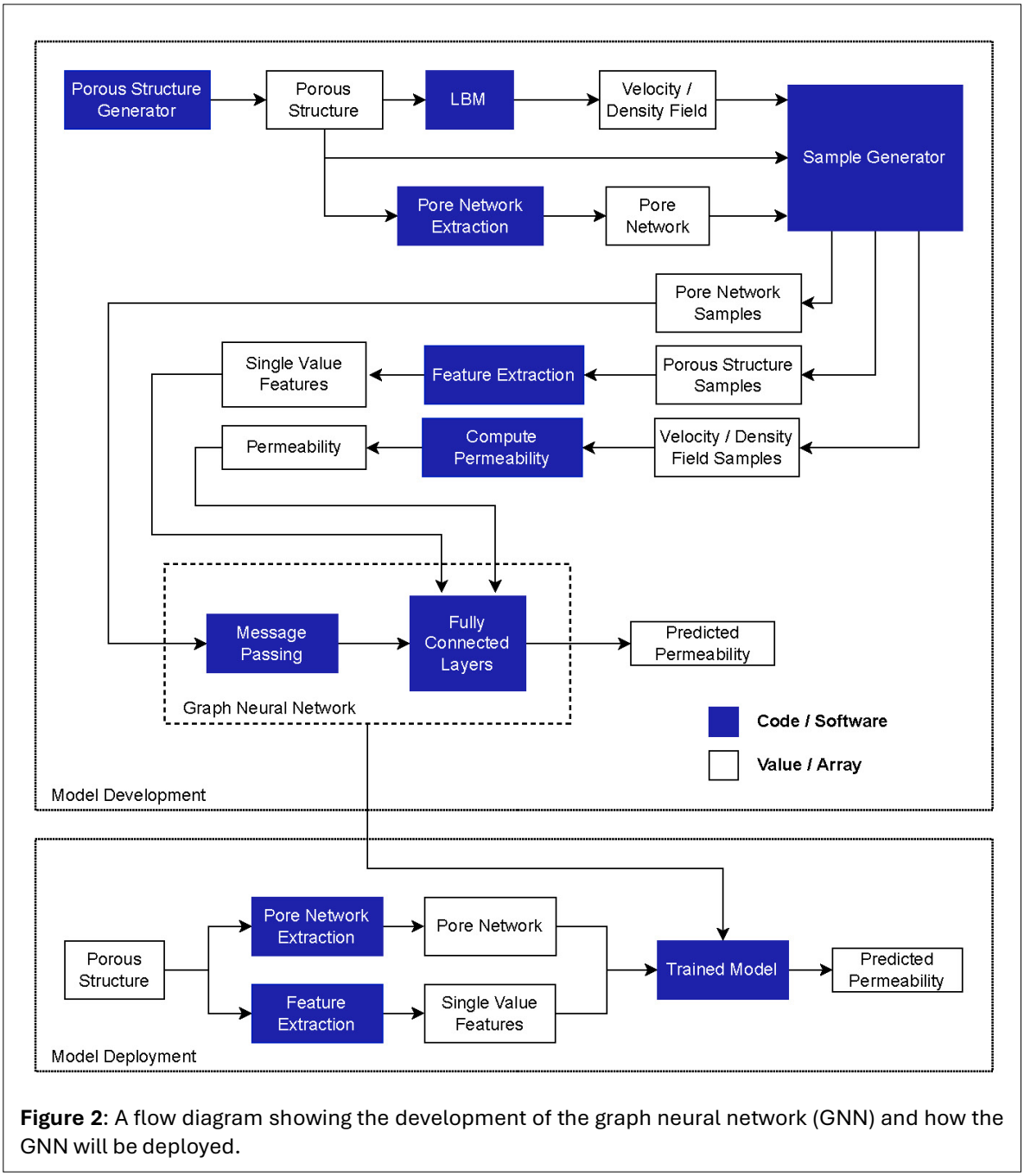

Figure 2: A flow diagram showing the development of the graph neural network (GNN) and how the GNN will be deployed.

2. METHODOLOGY

2.1. Method Overview

This section outlines the process for predicting the permeability of porous media using a GNN. **Figure 2** provides an overview of the model development framework and its deployment. Sections 2.2 to 2.6 provide details of each part of the framework.

2.2. Generating Structures

Training machine learning models to predict properties of porous media typically requires thousands of samples (3, 68, 76, 79). However, obtaining this dataset from real, virtually reconstructed porous media samples alone is difficult because samples can take hours to image with techniques such as X-ray tomography (42). Therefore, to test the method, a dataset was produced comprising randomly generated porous media. Several generation techniques exist in the porous media literature, including sphere packing (76, 83), cylinder packing (63), and the quartet structure generation set (QSGS) method (78, 79),

each with their own benefits and drawbacks. In this work, structures generated with fractal noise from the PoreSpy Python library (30) were used. This allowed for random, realistic structures with varying properties to be generated quickly and cheaply, with each structure taking just seconds to generate. Non-real structures were used because testing the method does not necessarily require real porous media samples. Training a model on fractal noise structures allows for a proof of concept of the approach, which can later be applied to more complex, real structures. The fractal noise structures were selected over other types such as packed spheres because they have more similarities such as a rough surface.

The resulting structure is a 3D array of values between zero and one. Binary porous structures were then generated by applying a threshold to the noise, which roughly corresponds to the porosity of the final

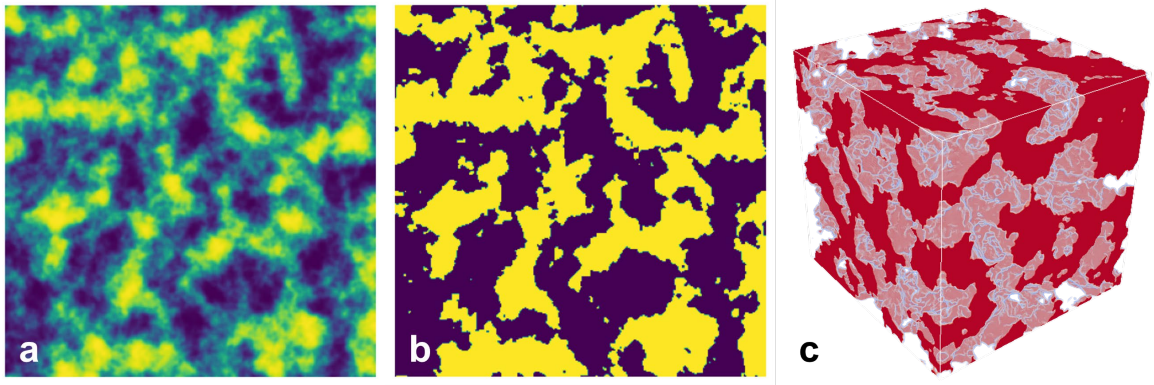

Figure 3: Example fractal noise porous structures: **a)** a 2D slice taken from a 3D fractal noise image, **b)** a 2D slice taken from a 3D binary image, and **c)** a voxelized 3D structure.

structure. Three example structures are shown in **Figure 3**, with **Figure 3a** showing a 2D slice of a 3D fractal noise image, **Figure 3b** showing a 2D slice taken from a 3D binary structure, where the light regions are the solid and the dark regions are the pores, and **Figure 3c** showing a 3D voxelized structure. Several parameters influence the properties of the structures; these include frequency, gain, and number of octaves of the noise (30), as well as the threshold applied to the noise. A Latin hypercube (LHC) design (56) was used to produce a set of structures with semi-random parameters over the entire range of possible parameters, ensuring the whole sample space is explored evenly, and improving on methods such as random or grid search. Three datasets were generated: one for training, one for validation and one for testing.

Typically, cross-validation is used to develop the model and identify the optimal hyperparameters (5). In this approach, the training and validation sets are derived from the training dataset, while a separate third dataset is reserved for final model testing. This is not possible here due to data leakage issues that arise when extracting validation samples from the training dataset. The sample extraction technique used is a form of data augmentation, a method used to enhance a dataset by producing new samples from the existing samples (72). If augmented samples from the same original structure are placed in both training and testing datasets, it can lead to seemingly better model accuracy, but the model will not generalize well to new data, which is a known issue in the CNN literature (82). Since the data augmentation techniques are similar to those used in the CNN literature, unique structures generated with the porous media generator (**Fig. 2**) were required for the training, validation, and testing datasets to avoid data leakage.

2.3. Feature Extraction

2.3.1. Structure Properties

A variety of physical structure properties can be extracted from the voxelized structures which can be used as input features for the fully-connected section of the GNN. Porosity can be obtained quickly by taking the volume fraction of the pore space in the total volume. Another useful property is the specific

solid surface area, which describes the surface area per unit volume, where the surface area is calculated from a mesh of the structure. The specific solid surface area was computed with the mesh for the Carman-Kozeny equation, but was approximated by using the fraction of solid voxels connected to pores as a proxy for the surface area for the GNN input. This provides a good, scale-independent representation for the GNN, without needing to compute the surface area from the mesh. Given the porosity and specific solid surface area, an estimate of the permeability can be found with the Carman-Kozeny equation (12, 43). The permeability obtained from this equation does not perform well on heterogeneous structures (61), however, it is useful to compare to the LBM simulations for more homogeneous structures. The Carman-Kozeny equation can take a variety of forms depending on the application. **Equation 1** shows a widely cited version based on the porosity and specific solid surface area (37, 67, 71, 75). The permeability obtained from the Carman-Kozeny equation was compared to the permeability obtained from the LBM and the GNN.

$$k = \frac{\phi^3}{k_{KC} S_0^2 (1-\phi)^2} \tag{1}$$

In **Equation 1**, k is the permeability in m^2, ϕ is the porosity, k_{KC} is the Kozeny constant, which is dimensionless, and S_0 is the specific solid surface area in m^{-1}. The Kozeny constant is typically given a value of 5 (12), which was also selected for this work. The porosity and specific solid surface area are macroscopic properties of the structures, however, heterogeneities in the structures will not be sufficiently represented by the macroscopic properties. As discussed by Nishiyama and Yokoyama (61), the critical pore diameter (defined as the maximum sized sphere that can pass through the porous structure) correlates well to the permeability, but is difficult to determine. For this reason, the LBM was used to determine the ground truth for the permeability of the structures, despite requiring significant computational cost.

Another common property used with the porous media field is the tortuosity, which describes the ratio of the free path between two points compared to the shortest distance between the two points (25). While there are versions of the Carman-Kozeny equation that include the tortuosity (71), there is significant ambiguity around its calculation with a range of methods providing different values that are weakly correlated and not interchangeable (85). This implies the physics behind these approaches is also inconsistent and therefore it is best avoided. Furthermore, representing the pore network as a graph effectively allows the data required to compute the tortuosity to be encoded into the adjacency matrix, meaning the tortuosity is not required as a feature.

2.3.2. Pore Network Extraction

Graph data structures, comprising an adjacency matrix and a feature matrix, extracted from the pore networks were used to train the GNN. The pore networks were extracted with PoreSpy (30) using a watershed segmentation algorithm (31). The PoreSpy implementation was chosen due to its speed and the variety of properties, such as the pore volume, diameter and surface area or connection length and diameter, which can be passed to the GNN. For visualization purposes, a 2D example of the extracted pore network is shown in **Figure 1**, however the 3D version works in the same way.

The adjacency matrix is a square matrix of size $M \times M$ where M is the number of pores in the pore network. Information about the edges can also be encoded in the adjacency matrix by weighting the connection values with edge features, known as a weighted adjacency matrix (15). The feature matrix is a 2D matrix of size $M \times N$ where N is the number of features. These features represent the properties of each pore.

2.3.3. Feature Selection

Extracting the pore networks with watershed segmentation provides node and edge feature matrices with several correlated features such as the pore inscribed diameter (the diameter of the largest sphere that can be placed in the pore) and pore equivalent diameter (the diameter of the sphere with a volume equal to the pore volume). Pearson correlation matrices were used to determine which of the pore and throat features to retain (9), as keeping too many correlated features increases model complexity and

can even lead to reduced model performance (33). Features with a correlation of greater than 0.5 can be regarded as strongly correlated (14). However, for this study, this threshold was increased to 0.8 to ensure only the most correlated features, such as inscribed pore diameter and equivalent pore diameter, were removed.

As the feature matrix is of size $M \times N$, any number of pore features can be selected, with the only cost being a slight computation time increase for more features. Many of the simple graph convolution layers used in GNNs only allow weighted adjacency matrices (41), with more computationally expensive layers allowing multiple edge features (27). The total length of each throat was taken as the single edge feature used to weight the adjacency matrix. This was because other throat features, such as diameter, are more closely related to the pore features, whereas the length is completely independent. This meant that the entire structure can be represented with a pore network and a few structure properties, which is much smaller than the voxelized structure required for the CNN.

2.4. The Lattice Boltzmann Method

The LBM has been used for a variety of applications including multiphase flows (51), moving geometries (49), heat transfer (74), as well as flow through porous media (20, 76, 79). As mentioned in the Introduction, traditional CFD can be used to perform porous media simulations, however, complex meshes must be generated to adequately represent the complex geometries found with porous media (58). They often require further refinement closer to the surfaces to provide sufficient surface definition (7), further increasing the mesh size. Due to the voxelized domain, the LBM is very well suited to porous media simulations, removing the need for these complex meshes (47), and is not subject to continuum modeling constraints (34). As the generated structures are binary 3D arrays, they are in a voxelized format, meaning a simulation technique which can deal with voxelized domains is ideal.

2.4.1. Simulating Flow Through Porous Media

The porous media simulations were performed with Palabos (50), an open-source C++ library capable of performing highly parallelized simulations. The chosen lattice was a D3Q19 lattice—a 3D lattice with 19 discrete velocities per node—selected for its balance between computational cost and accuracy (48). The Bhatnagar-Gross-Krook (BGK) collision operator was chosen for its simplicity (48), though other, more complex, collision operators are available (45). A uniform pressure gradient was imposed between the inlet and outlet which drives flow through the structure. A no-slip impermeable boundary condition was applied to the edges of the domain adjacent to the direction of flow, and the full-way bounce-back scheme was used for the fluid-solid collisions within the structure that occur on the boundary lattice nodes between the solid and the fluid.

2.4.2. Computing Permeability

Once the simulation had been completed, the permeability was calculated from the component of the velocity field in the direction of flow using a rearrangement of Darcy's law, shown in **Equation 2**.

$$k = \frac{q\mu L}{\Delta P} \tag{2}$$

Here k is the permeability, q is the average velocity component in the direction of flow, μ is the lattice viscosity, L is the length of the domain in the direction of flow, and ΔP is the change in pressure in the direction of flow, all in lattice units.

2.5. Producing a Dataset

As mentioned previously, obtaining a dataset large enough to train the GNN from entirely real data is expensive, so non-real structures were generated to mimic the properties of real structures. It was also mentioned that the simulations take a long time to complete, which is the reason a data-driven approach for determining the porous structure properties is useful. While the structure generation is quick for the non-real structures, the LBM simulations would require significant computation time. An augmented dataset can be generated by extracting several smaller, overlapping samples from a single larger structure, providing the samples are representative of the larger structure. The size at which this is true

is known as the representative elementary volume (REV) scale (32), explained further in Section 2.5.1. This allows for many more partially unique structures to be extracted from a single simulation, reducing computation time for a given dataset. The structures are only partially unique because each sample extracted from a single structure may have some overlap with other samples.

Computing the permeability of the samples requires an extra step because the pressure drop between the inlet and the outlet of the sample is not known. The pressure drop can be calculated from the inlet and outlet density using **Equation 3** (46).

$$\Delta P = c_s^2(\rho_{in} - \rho_{out}) \tag{3}$$

Where ΔP is the change in pressure between the inlet and outlet, ρ_{in} and ρ_{out} are the average inlet and outlet densities, obtained from the density field, and c_s is the speed of sound in the lattice, given the value $\sqrt{1/3}$, all in lattice units.

2.5.1. Representative Elementary Volume

The REV-scale refers to the volume at which the macroscopic properties of a sample are the same as the macroscopic properties of the structure from which the sample was extracted (32). When randomly sized samples are extracted from a structure, there is more spread in the permeability for smaller samples,

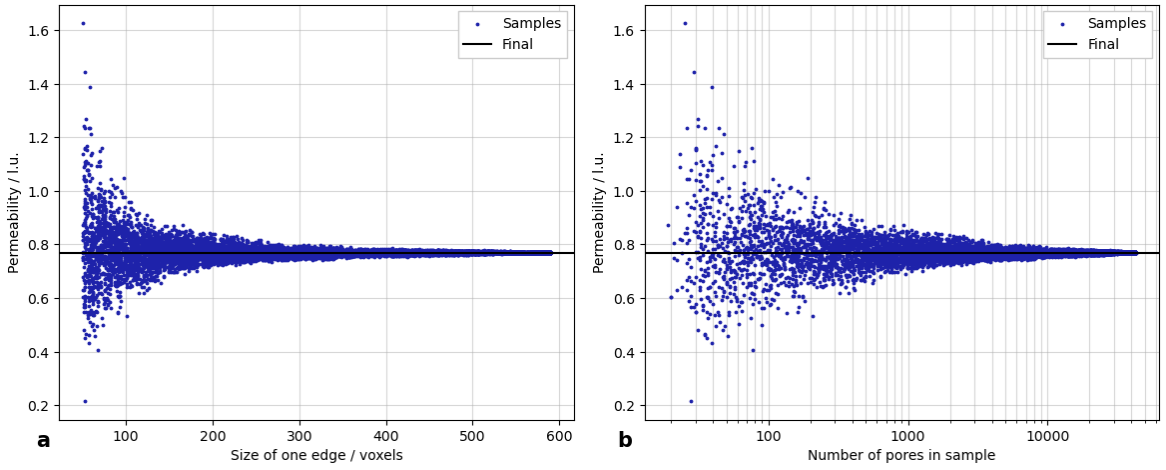

Figure 4: Trumpet plot showing the permeability of random samples extracted from a larger structure as a function of **a)** the size of the structure in voxels and **b)** the number of pores in the pore network.

meaning each of the small samples are not representative of the structure they were sampled from. **Figure 4a** shows the permeability of the cubic samples as a function of the length of one side and **Figure 4b** shows the permeability of the same samples as a function of the number of pores in the pore network. The representative scale for a given structure can be quantified as the point at which the spread of permeabilities is within a given threshold. It can be given in terms of the volume (REV) or the pore count, denoted as the representative pore count throughout this work. For example, a threshold of $\pm$10% would define the representative scale to be the point at which all the samples are within a range of $\pm$10% of the permeability of the structure the samples were taken from. A lower threshold value would be more accurate, but each sample would need to be larger, while a higher threshold would be less accurate but would require smaller samples, reducing computation costs. The chosen value was $\pm$10% for this work.

2.5.2. Extracting Samples

A pore network was extracted, and the LBM simulation was performed on each of the generated structures. From each structure, 500 smaller, overlapping samples were extracted, thereby reducing the computational cost of the dataset. Extracting samples of a fixed size from each structure results in pore networks with comparable numbers of pores and similar permeabilities. This inadvertently caused the GNN to use the number of pores as a predictor, which was not intended and led to degraded model performance. Therefore, the samples were randomly sized between a range of 100 and 400 voxels per

dimension, leading to varying pore network sizes. A margin was applied around the edges of the structures from which samples could not be extracted. This was to reduce the effects of the boundary conditions from the LBM simulations on the permeability value. The margin size was defined based on the point at which the permeability stops changing as the margin increases, indicating the effects of the margin were negligible.

2.5.3. Dataset Processing and Structure

The structure features, the feature matrix and the permeability were all scaled before training to ensure accurate and stable training (39). The feature matrix and the structure features were normalized to have values between zero and one, ensuring the features with large values are weighted equally to features with small values. The adjacency matrix was scaled with symmetric normalization, explained further in Section 2.6.1. The permeability for the samples spanned multiple orders of magnitude (in lattice units), and therefore a natural log transform was applied to the permeability, ensuring errors in the prediction of samples with high permeability are penalized equally to the samples with low permeability. Both the validation and test datasets were scaled using the values obtained from the training dataset to ensure independence, and thus avoid information from the training dataset influencing the testing performance (38), which would lead to data leakage.

Each pore network must be of the same size to be passed to the GNN. To ensure this, the sample with the maximum number of pores, M_{max}, was identified, and pore networks with fewer pores were zero-padded to match this size. The extra zeros do not have a significant effect on the training, just as two pores that are not connected would contribute little to the training.

2.6. Graph Neural Network

The neural network was developed and trained using PyTorch (66) and PyTorch Geometric (23). This allowed large GNNs to be trained quickly, using multiple GPUs on a HPC cluster. The GNN allows the model to be independent of the size of the structure (in terms of voxels), compared to CNNs which take in images of the same size (69), limiting the model structures to a fixed structure size. The GNN takes in a graph and encodes it into a new feature matrix with information from the original feature matrix and the connections from the adjacency matrix. The GNN was split into two parts: a set of message passing layers and a set of fully connected layers.

2.6.1. Message Passing

The goal of the message passing is to learn node embeddings from the node features (24). After each message passing layer, each node gains information about its neighboring nodes. Multiple message passing layers can be joined in series, with each layer providing information about the connections further away from a given node. However, using too many message passing layers can lead to over-smoothing of the network, where each of the resulting features have similar values (11). **Figure 5** illustrates how the message passing layers pass information from a node to its neighbors after each layer. Initially, the feature matrix only contains the selected pore features, with no information about the pore connections. Considering just the pore highlighted with an 'x', the feature values for that node are

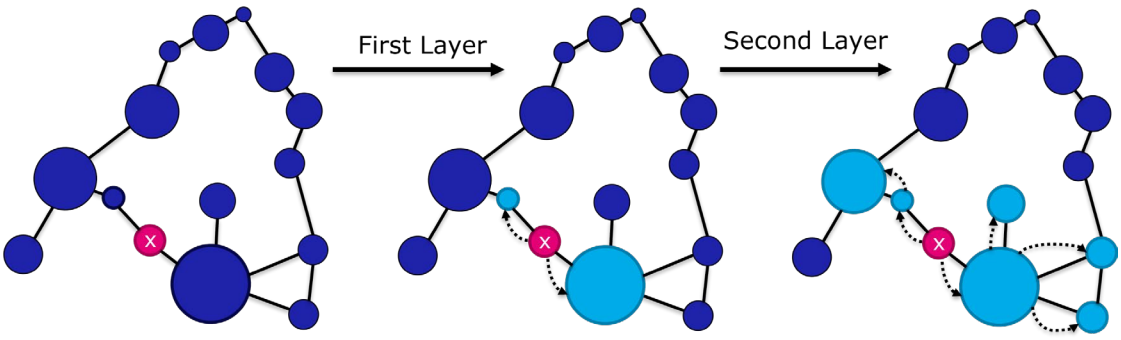

Figure 5: How information is passed from one node to its neighboring nodes after each message passing layer.

just the pore features. After a single message passing layer, the feature values of the pore highlighted with an 'x' have information about its two neighbors, highlighted in light blue.

Equation 4 shows the main equation describing the message passing (41),

$$F_{n+1} = \sigma(\widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}F_n W_n) \quad (4)$$

where σ is an activation function and W_n is a matrix of trainable parameters called the weight matrix. $\widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}$ is the symmetric normalized Laplacian matrix (65) which can help to eliminate some training instability compared to some other normalization techniques (41), while also maintaining the symmetry of the adjacency matrix, which is important because asymmetry of the adjacency matrix would imply a directed graph, which the pore network is not. F_n and F_{n+1} are the feature matrices after n and $n+1$ message passing layers. Several activation functions were tested including TanH and ReLU, but they lead to unstable training; therefore, the leaky ReLU was used. The size of the weight matrices in the message passing layers and the number of message passing layers are hyperparameters of the model and need to be optimized. This is to ensure enough information from the feature matrix and adjacency matrix is passed to the fully connected layers without causing the over-smoothing effects.

2.6.2. Fully-connected Layers

Following message passing, the node embeddings are flattened and combined with the single value features (porosity, specific solid surface area and sample size) and passed to the input of a fully-connected neural network. There are several architectural choices to be made for the fully connected layers; these include the number of hidden layers and the number of nodes per layer (28). As with the message passing layers, the leaky ReLU activation function was chosen because it proved to be the most stable during the training process.

The output of the fully-connected layers was a single value which represented the prediction of the natural log of the permeability. The mean squared error (MSE) (57) was used as the criterion for assessing the model performance.

2.6.3. Hyperparameter Tuning

Tuning the hyperparameters in any supervised machine learning model with several tunable parameters is important for finding the optimal model (6). Manually selecting hyperparameters is a nearly impossible task for problems with potentially millions of hyperparameter combinations. Because of this, Optuna (1) was used to determine a set of optimal hyperparameters. Optuna employs efficient sampling algorithms to select new configurations of hyperparameters based on the previous training runs. It can also prune

Table 1: Summary of the hyperparameter options for Optuna.

Hyperparameter	Options for Optuna
Number of inner message passing layers	0-3
Size of each message passing layer	2, 4, 6, 8, 10 or 12
Size of final message passing layer	1 or 2
Number of hidden fully-connected layers	2-6
Size of each fully-connected layer	32, 64, 128, 256, 512, 1024, 2048
Optimizer	SGD, Adam, AdamW, Adamax
Learning rate	0.001-0.1 (log uniform)
Learning rate step	10-100
Learning rate decay	0.8-1
Optimizer beta values	0.75-1

SGD: Stochastic Gradient Descent

poorly performing trials, and thus was chosen over other methods such as grid search or random search to reduce the computation costs of the hyperparameter search (2, 22).

Table 1 shows a summary of the hyperparameter options for the Optuna hyperparameter tuning. The number of message passing layers had a range of 1-4 to avoid over-smoothing. The number of nodes in each layer was kept relatively low compared to the number of nodes in the fully-connected layers due to memory requirements. The size of the final message passing layer was capped at two nodes because it is flattened and passed to the fully-connected layers, so large values lead to further memory issues. The number of fully-connected layers had a range of 2-6, and each layer had a width of between 32 and 2048 nodes. This allows both the depth and width of the fully-connected neural network to be explored in the hyperparameter search. The stochastic gradient descent (SGD) and Adam (40) optimizers were included in the hyperparameter search as they are some of the most widely used. Variations of Adam that have been used for GNNs were also tested, including AdamW (64) and Adamax (19). Finally, the learning rate, including the learning rate scheduler hyperparameters, were included in the hyperparameter search, along with the beta values (28), which are parameters of the Adam, Adamax and AdamW optimizers.

A hyperparameter study was set up and allowed to run until a given number of successive trials were pruned. The trials were trained on the entire training dataset and tested on the entire validation dataset. Each trial was allowed to run for up to 250 epochs, unless the trial was stopped early. The median pruner was chosen as the early stopping algorithm, since it allows several warm-up steps before pruning is checked. This is useful because the training process of the GNN can be initially unstable before stabilizing into a more typical learning curve. The selected sampler was the tree-structured Parzen estimator, chosen to accommodate the relatively small number of trials imposed by the long training time of the GNN. The optimization criterion for the Optuna study was based on the MSE of the validation dataset at the end of the training process.

2.6.4. Model Testing

Once the hyperparameter study had been completed, a final model was then trained using the optimal hyperparameters. This model was trained for up to 500 epochs as the computational costs of a single training run are less significant than the combined hyperparameter study trials. Training for longer than this resulted in overfitting of the model, leading to significantly better performance on the training dataset, with no improvement on the testing performance. The third, unseen testing dataset was used to test the performance of this model to ensure the testing dataset was independent and to avoid a biased estimation of generalization.

Finally, the trained model was compared to the Carman-Kozeny equation. A set of 15 new structures were generated with porosities ranging from 0.2 to 0.8. These structures were the same size as the structures used in the other datasets, and every other generation parameter was kept constant. An LBM simulation was performed on each of the structures to obtain the velocity and density fields. A 10% margin region was placed around the structures, and 100 randomly sized samples were extracted from random locations outside of the margin regions. The pore network from each sample was passed to the trained model, the porosity and specific solid surface area were used to obtain the permeability from **Equation 1**, and the velocity and density fields were used to compute the permeability from the LBM simulations to compare the three methods.

3. RESULTS AND DISCUSSION

3.1. Dataset Generation

3.1.1. Generating Structures

Three independent sets of 500 × 500 × 500 voxel structures were generated: one for training with 200 structures, one for validation with 50 structures, and one for testing with 50 structures. The three sets of parent structures were kept separate to ensure there was no data leakage caused by overlapping samples.

3.1.2. Pore Network Feature Selection

Correlation matrices were produced for both the pore and throat features, and the features with a correlation of more than 0.8 were removed from the dataset, simplifying the model. The selected pore features were the three coordinates, extended pore diameter, and pore region volume, while the selected throat feature was total length.

3.1.3. LBM Simulations

The LBM simulations were performed on an HPC cluster using a 44-core CPU. The simulation results included a density scalar field and velocity vector field. The density field and the x-component of the velocity field were passed to the sample generator. This was the same code that was used to sample the voxelized structures as highlighted by the flow diagram in **Figure 2**. **Figure 6** shows an example pressure gradient and 3D velocity field from the LBM simulations. The resulting dataset had permeabilities spanning orders of magnitude between 10^{-3} and 10^{2} in lattice units.

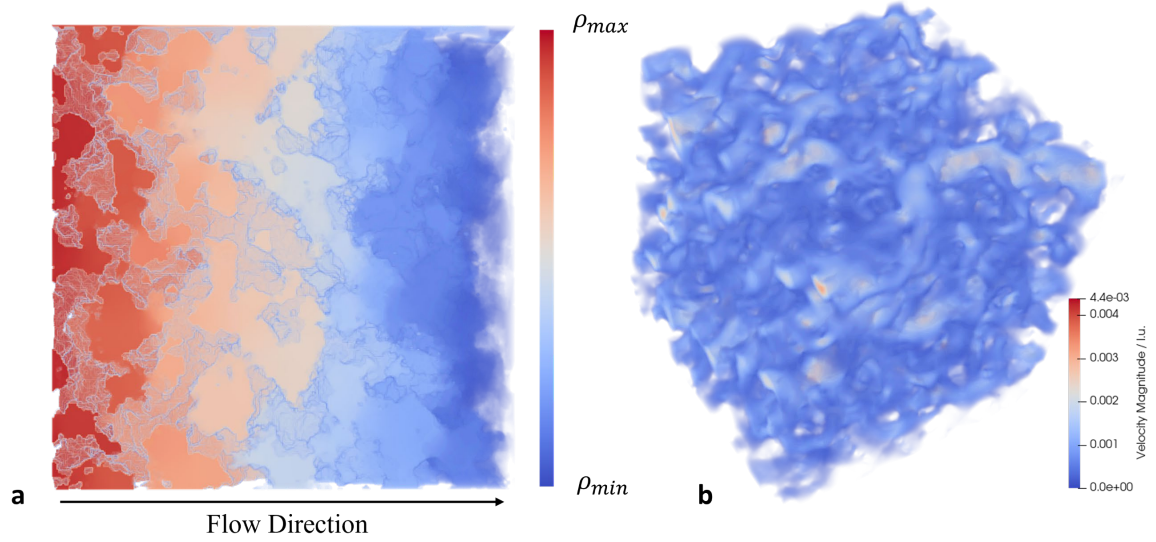

Figure 6: Visualization of some lattice Boltzmann method simulation results, with **a)** showing the density field across the direction of flow and **b)** showing the 3D velocity field.

3.1.4. Representative Scale Analysis

The representative pore count was found for a range of structures of increasing noise frequency (which is equivalent to a decrease in average pore size of the pore network) in terms of both number of voxels and number of pores in the pore network. **Figure 7** shows how the representative scale changes as a function of the noise frequency in terms of both volume (voxels) and pore count. There is a decrease in the required voxels for the sample to be representative, but the required number of pores in the pore network does not show a clear trend and fluctuates around 1200 pores.

Figure 7 shows the benefits of defining the representative scale based on the number of pores rather than the number of voxels. If the representative volume was used, every sample, regardless of the noise frequency (and therefore average pore size), would need to have over 300^3 voxels, whereas only 150^3 voxels are needed for the higher frequency structures. This increases the size of the dataset and slows model training. Instead, using the representative pore count provides a more consistent value of 1000-1400 pores per sample across the entire range of noise frequencies. Using number of pores also allows for non-cubic samples to be used providing they have the required number of pores in the pore network. Based on the results in **Figure 7**, the minimum number of pores required in a pore network sample was chosen to be 1500 and any sample with fewer than 1500 pores was removed from the dataset.

3.1.5. Samples Dataset

From each of the 200 training structures, 500 samples were extracted, resulting in a total of 100,000 training structures. The same process was repeated for the validation and testing datasets to produce

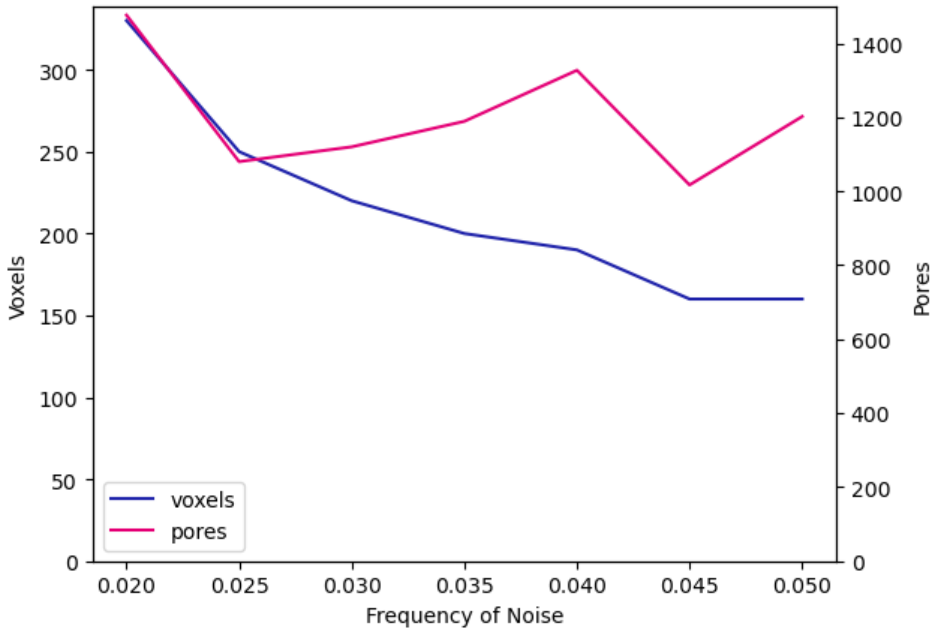

Figure 7: Representative scale analysis in terms of both number of pores in the pore network (pink) and number of voxels in the structure (blue).

25,000 validation samples and 25,000 testing samples. During sample extraction, samples with fewer than 1500 pores were discarded because they were not representative samples. Similarly, samples with more than 6000 pores were discarded to ensure the size of each sample was not too large, which would increase computation costs and therefore training time.

3.2. Graph Neural Network

3.2.1. Selecting the Optimal Model

The Optuna study was allowed to run until a significant number of successive trials were pruned. At the end, a total of 178 trials were completed, of which 46 were successful. The pruned trials were still useful for the study as it showed the combinations of hyperparameters that lead to poor performance. The study was not continued further because the improvements were minimal. The optimal hyperparameters are shown in **Table 2**. The fully-connected layer input had 6,005 nodes, which is based on the product of the number of features following the message passing layers and the number of pores in the pore networks, plus the five structure features.

Table 2: Summary of the optimal hyperparameters from the Optuna study.

Hyperparameter	Optimal Value
Message passing layer	[2, 1]
All layer sizes	[6005, 512, 256, 1]
Optimizer	AdamW
Initial learning rate	0.00971
Learning rate step	19
Learning rate decay	0.827
AdamW Beta Values	[0.925, 0.829]

3.2.2. Training the Optimal Model

Once the optimal hyperparameters were found, a final model was trained with the full training dataset and tested on the unseen testing dataset. This model was allowed to run for a total of 500 epochs, taking an average of 16.02 seconds per epoch to train with four 16 GB Nvidia T4 GPUs and had 3,206,672 trainable parameters. During the training process, the results and checkpoints of the models were stored every 10 epochs, allowing for intermediate models to be retrieved. Beyond 20 epochs, the testing MSE started to increase while the training MSE continued decreasing, indicating that the model was beginning to overfit. The learning rate for the optimal model was relatively high, which led to the quick training. **Figure 8a** shows the training and testing MSE for each epoch

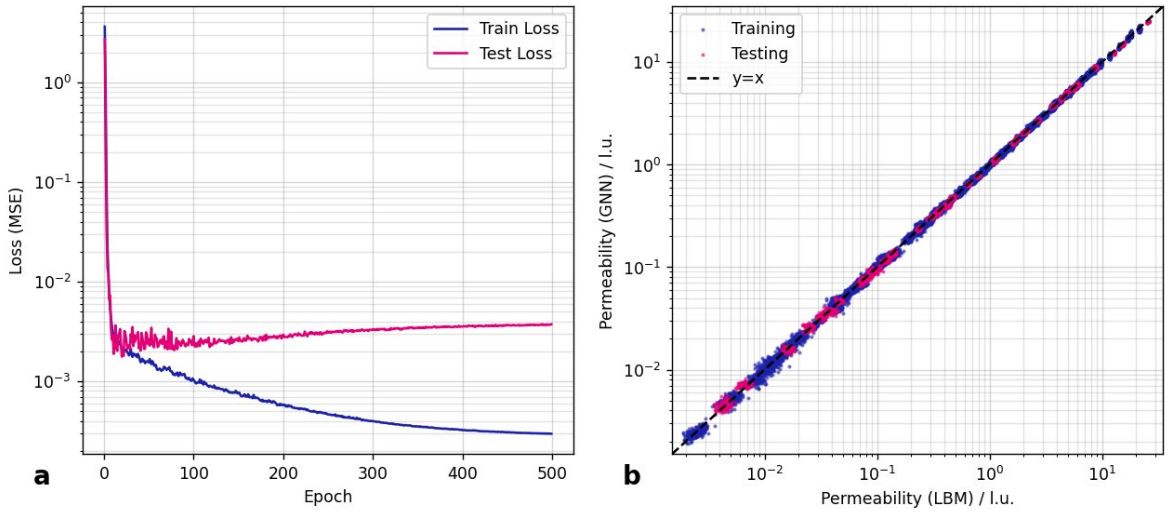

Figure 8: a) The training and testing MSE per epoch of the full training process and **b)** the parity plot for the final chosen model after 20 epochs.

during the training process and **Figure 8b** shows the parity plot for both training and testing data for the optimal model after 20 epochs of training.

The model showed good performance on both the training and testing dataset over nearly four orders of magnitude, showing the trained model can generalize well to the unseen testing dataset. The learning curves appear noisy, however, the amount of noise is exaggerated by the log-scale on the y-axis. Noisy learning curves are not uncommon for deep learning training, with several examples of well-performing models with noisy learning curves being shown in the literature (3, 76). The R^2 of the training data was 0.999, scored on the natural log of the permeability. However, using the R^2 as a metric for performance for this model is misleading because the parity plot is on log-log axes and the data spans several orders of magnitude. An alternative metric for model performance is a direct comparison of the GNN against the Carman-Kozeny equation. The MSE and MAE were the chosen metrics, and they were calculated on the natural log of the permeability for both models as that was what the GNN was aiming to predict. The comparison is shown in **Figure 9**.

The GNN showed good agreement with the LBM over the entire porosity range and for permeabilities that span nearly three orders of magnitude. The error bars on the GNN prediction were small, highlighting the accuracy of the GNN. The tight error bars on the LBM results also show the 100 samples used are representative of the parent structures used for testing. The GNN outperformed the Carman-Kozeny equation in terms of both MSE and MAE, with an MSE of 0.00190 and an MAE of 0.0302 for the GNN, compared to an MSE of 1.125 and an MAE of 0.783 for the Carman-Kozeny equation. The Carman-Kozeny equation overpredicted the LBM permeability over the entire range of structures, more significantly at the lower porosities, compared to the GNN which matches the LBM permeability over the entire range. There have been several formats of the Carman-Kozeny equation developed based on properties including porosity, surface area, mean pore size, and tortuosity (71). The chosen version for this comparison was based on the specific solid surface area because the fractal structures used to train the models have a high surface area. Despite this, the Carman-Kozeny equation still overpredicted the LBM simulations, suggesting the surface area parameter is not sufficiently capturing the structure properties, especially at lower porosities. Furthermore, using a data-driven approach such as the GNN for permeability estimation has the potential to be applied to any given porous media type included in the training set, which is not the case for the Carman-Kozeny equation.

These results show the capability of the GNN to handle representative samples. The GNN trained for this work was significantly smaller than the upper limit of GNN size on the same hardware. Large batch sizes of 1024 were used, while still not fully utilizing the GPU memory. The GNN was built to handle samples to a maximum of 400^3 voxels, however, the REV-scale of some real porous media samples can often be

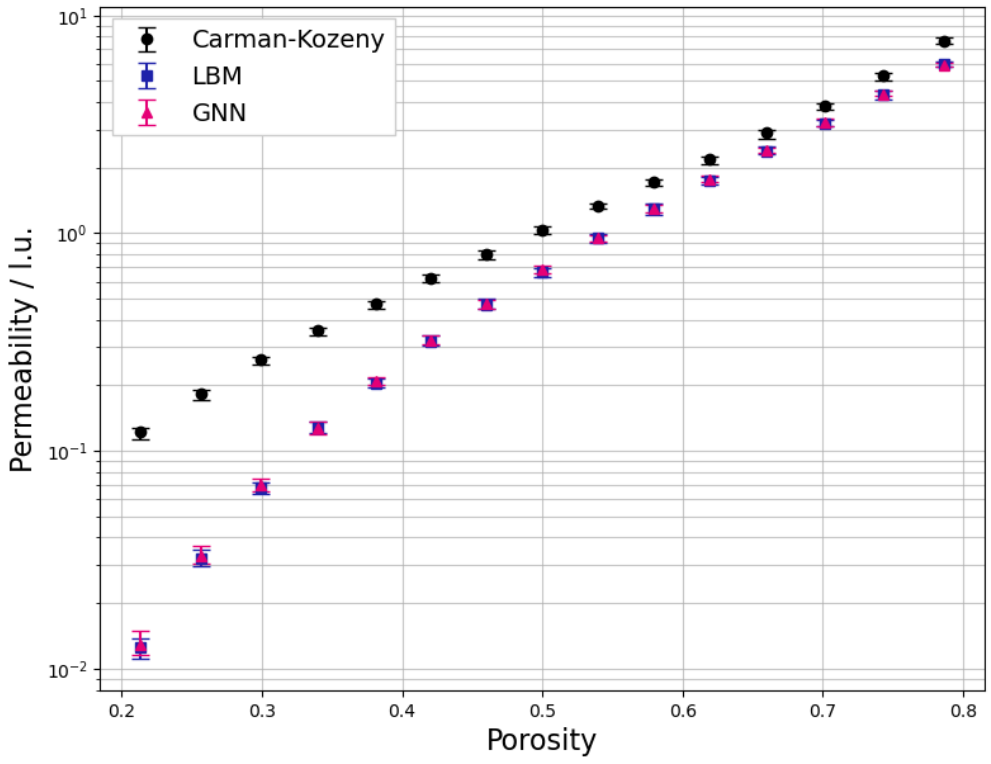

Figure 9: A comparison of the permeability obtained from the graph neural network (GNN), the lattice Boltzmann method (LBM) and the Carman-Kozeny equation for structures of varying porosity.

larger than this (73). Despite this, given the option to lower batch size to allow for larger models, we believe the GNN will be scalable enough to handle porous media samples with a much larger REV-scale.

Finally, the GNN was able to make predictions of the permeability significantly faster than the LBM simulations, even when including the time required to extract the pore networks. **Figure 10** shows a comparison of the LBM simulation time, the GNN inference time and the GNN inference time with the pore network extraction time included for five randomly selected porous media samples of the same type as those used in the training dataset. For the purposes of the comparison, a four-core CPU was used for each example. The GNN inference time is extremely quick for each sample at around 8 ms, compared to the average LBM simulation time of around 37 minutes. It should also be noted that the LBM convergence criteria were relatively relaxed for this work to allow for faster and cheaper model development. Application of the method to real structures would benefit from more strict convergence criteria, which would further increase the LBM simulation time, without affecting the GNN inference time. The time for the GNN inference and the pore network extraction was on average under 4 minutes - nearly 10 times faster than the LBM simulation, showing the benefits of the GNN approach over the LBM simulations. Furthermore, analysis of porous media often includes some type of pore network extraction to gather information such as pore size distribution, so it is often already computed, in which case the GNN is able to provide a near-instant prediction of the permeability.

4. CONCLUSIONS

In this work, a GNN was trained to predict the permeability of artificial porous media samples using pore networks extracted from within the structures, as well as some physical structure properties, such as the porosity and specific solid surface area. A representative pore count study was performed, showing that samples with at least 1500 pores were representative of the structures the samples were extracted from. This allowed for a GNN to be trained that is large enough to provide accurate predictions of the structures of interest. The permeability data used to train the model was determined using LBM

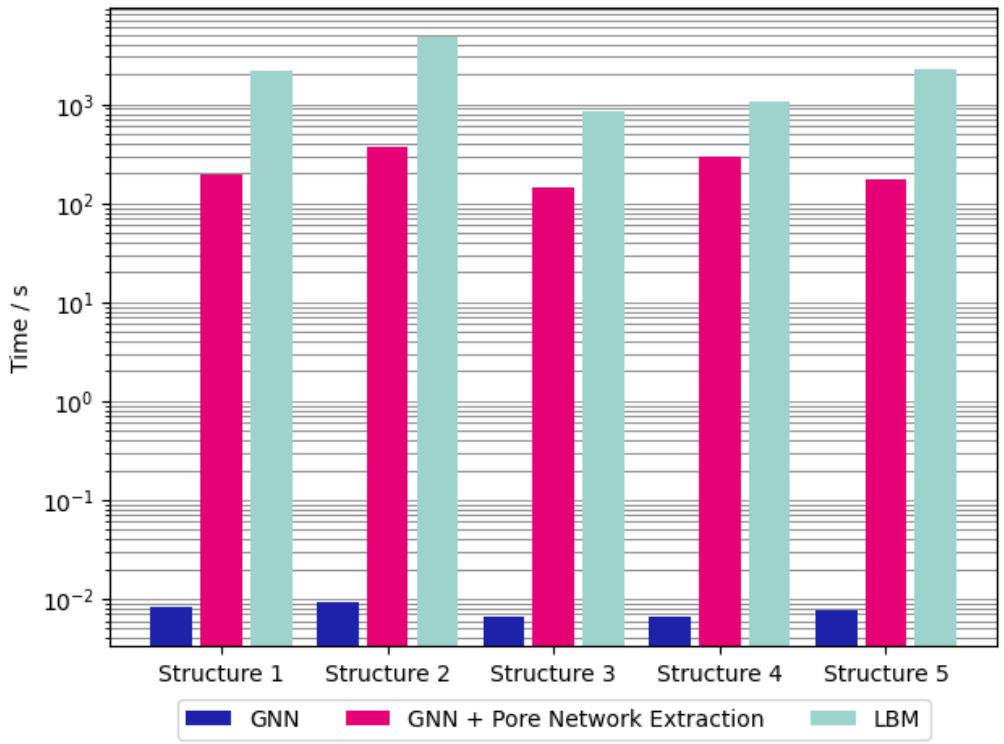

Figure 10: A comparison of the lattice Boltzmann method (LBM) simulation time vs. the graph neural network (GNN) inference time and the GNN inference time with the pore network extraction time included.

simulations. The GNN was able to make fast and accurate predictions of the permeability, showing a good agreement with the LBM simulations, outperforming the commonly used Carman-Kozeny equation, with the GNN having an MSE of 0.00190 and an MAE of 0.0302, and the Carman-Kozeny equation having an MSE of 1.125 and an MAE of 0.783 when compared to the LBM simulations. The GNN inference time was several orders of magnitude faster than the LBM simulations, with the same computational resources, and even with the pore network extraction included in the GNN time, it was still nearly 10 times faster than the LBM simulations.

While the GNN was able to outperform the existing correlations and showed a good agreement with the simulations, it should be noted that the model was trained and tested on ideal porous media samples. Further work is needed to expand on the method and apply the GNN to real porous media samples where the representative scale is significantly higher due to heterogeneities often found within real porous media samples. Despite this, the GNN is a promising tool for predicting porous media properties, and provides a more scalable approach than existing methods, such as the 3D CNN, due to the scale being tied to the pore count of the pore network rather than the voxel count of the porous structure.

STATEMENTS AND DECLARATIONS

Acknowledgements

We would like to acknowledge Dr. Estefania Lopez-Quiroga for her helpful comments on the manuscript.

Author Contributions

Jack M I'Anson: conceptualization, data curation, formal analysis, investigation, methodology, software, visualization, writing - original draft. M. J. H. Simmons: methodology, funding acquisition, supervision, writing - review & editing. E. H. Stitt: methodology, supervision, writing - review & editing. R. W. Gallen: conceptualization, methodology, project administration, resources, supervision, writing - review & editing.

Conflicts of Interest

There are no conflicts of interest to declare.

Funding Received

Jack I'Anson is funded by a studentship from the EPSRC CDT in Formulation Engineering: Sustainable Structured Products (EP/S023070/1) and Johnson Matthey.

ORCID IDs

Jack M. I'Anson https://orcid.org/0009-0002-8284-5105
Mark J. H. Simmons https://orcid.org/0000-0002-0655-3744
E. Hugh Stitt https://orcid.org/0000-0002-2208-882X
Robert W. Gallen https://orcid.org/0000-0003-4875-498X

REFERENCES

1. Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2623–2631. https://doi.org/10.1145/3292500.3330701
2. Alibrahim, H., & Ludwig, S. A. (2021). Hyperparameter optimization: Comparing genetic algorithm against grid search and bayesian optimization. *2021 IEEE Congress on Evolutionary Computation (CEC)*, 1551–1559. https://doi.org/10.1109/CEC45853.2021.9504761
3. Alqahtani, N., Alzubaidi, F., Armstrong, R. T., Swietojanski, P., & Mostaghimi, P. (2020). Machine learning for predicting properties of porous media from 2D X-ray images. *Journal of Petroleum Science and Engineering*, *184*, 106514. https://doi.org/10.1016/j.petrol.2019.106514
4. Alzahrani, M. K., Shapoval, A., Chen, Z., & Rahman, S. S. (2023). Pore-GNN: A graph neural network-based framework for predicting flow properties of porous media from micro-CT images. *Advances in Geo-Energy Research*, 10(1), 39–55. https://doi.org/10.46690/ager.2023.10.05
5. Arlot, S., & Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4 (none). https://doi.org/10.1214/09-SS054
6. Bardenet, R., Brendel, M., Kégl, B., & Sebag, M. (2013). Collaborative hyperparameter tuning. In S. Dasgupta & D. McAllester (Eds.), *Proceedings of the 30th International Conference on Machine Learning* (Vol. 28, pp. 199–207). PMLR. https://proceedings.mlr.press/v28/bardenet13.html
7. Boccardo, G., Plato, L., Marchisio, D., Augier, F., Haroun, Y., et al. (2014, June). *Pore-scale simulation of fluid flow in packed-bed reactors via rigid-body simulations and CFD*. https://www.researchgate.net/publication/264897796_PORE-SCALE_SIMULATION_OF_FLUID_FLOW_IN_PACKED-BED_REACTORS_VIA_RIGID-BODY_SIMULATIONS_AND_CFD
8. Bryntesson, L. M. (2002). Pore network modelling of the behaviour of a solute in chromatography media: Transient and steady-state diffusion properties. *Journal of Chromatography A*, 945(1–2), 103–115. https://doi.org/10.1016/S0021-9673(01)01485-6
9. Buyrukoğlu, S., & Akbaş, A. (2022). Machine learning based early prediction of type 2 diabetes: A new hybrid feature selection approach using correlation matrix with heatmap and SFS. *Balkan Journal of Electrical and Computer Engineering*, 10(2), 110–117. https://doi.org/10.17694/bajece.973129
10. Cai, C., Vlassis, N., Magee, L., Ma, R., Xiong, Z., et al. (2021). *Equivariant geometric learning for digital rock physics: Estimating formation factor and effective permeability tensors from Morse graph*. https://doi.org/10.48550/ARXIV.2104.05608
11. Cai, C., & Wang, Y. (2020). *A note on over-smoothing for graph neural networks*. arXiv. https://doi.org/10.48550/ARXIV.2006.13318
12. Carman, P. C. (1997). Fluid flow through granular beds. *Chemical Engineering Research and Design*, 75, S32–S48. https://doi.org/10.1016/S0263-8762(97)80003-2
13. Chen, Y., Liu, J., Zhang, X., Qi, X., & Jia, J. (2023). Largekernel3d: Scaling up kernels in 3D sparse CNNS. *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 13488–13498. https://doi.org/10.1109/CVPR52729.2023.01296

14. Cohen, J. (1977). Differences between correlation coefficients. In *Statistical Power Analysis for the Behavioral Sciences* (pp. 109–143). Elsevier. https://doi.org/10.1016/B978-0-12-179060-8.50009-8
15. Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, Clifford. (2001). Representations of Graphs. In *Introduction to Algorithms* (2nd ed., pp. 527–531). MIT Press. https://mitpress.mit.edu/9780262531962/introduction-to-algorithms/
16. Cummings, D., & Nassar, M. (2020). Structured citation trend prediction using graph neural networks. *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 3897–3901. https://doi.org/10.1109/ICASSP40776.2020.9054769
17. Dai, M., Demirel, M. F., Liang, Y., & Hu, J.-M. (2021). Graph neural networks for an accurate and interpretable prediction of the properties of polycrystalline materials. *Npj Computational Materials*, 7(1), 103. https://doi.org/10.1038/s41524-021-00574-w
18. Dong, G., & Liu, H. (Eds.). (2018). *Feature engineering for machine learning and data analytics*. CRC Press.
19. Duval, A. A., Schmidt, V., Hernández-García, A., Miret, S., Malliaros, F. D., et al. (2023). Faenet: Frame averaging equivariant GNN for materials modeling. In A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, & J. Scarlett (Eds.), *Proceedings of the 40th International Conference on Machine Learning* (Vol. 202, pp. 9013–9033). PMLR. https://proceedings.mlr.press/v202/duval23a.html
20. Eshghinejadfard, A., Daróczy, L., Janiga, G., & Thévenin, D. (2016). Calculation of the permeability in porous media using the lattice Boltzmann method. *International Journal of Heat and Fluid Flow*, 62, 93–103. https://doi.org/10.1016/j.ijheatfluidflow.2016.05.010
21. Fan, W., Ma, Y., Li, Q., He, Y., Zhao, E., et al. (2019). Graph neural networks for social recommendation. *The World Wide Web Conference*, 417–426. https://doi.org/10.1145/3308558.3313488
22. Feurer, M., & Hutter, F. (2019). Hyperparameter optimization. In F. Hutter, L. Kotthoff, & J. Vanschoren (Eds.), *Automated Machine Learning: Methods, Systems, Challenges* (pp. 3–33). Springer International Publishing. https://doi.org/10.1007/978-3-030-05318-5_1
23. Fey, M., & Lenssen, J. E. (2019). *Fast graph representation learning with pytorch geometric*. arXiv. https://doi.org/10.48550/ARXIV.1903.02428
24. Flam-Shepherd, D., Wu, T. C., Friederich, P., & Aspuru-Guzik, A. (2021). Neural message passing on high order paths. *Machine Learning: Science and Technology*, 2(4), 045009. https://doi.org/10.1088/2632-2153/abf5b8
25. Ghanbarian, B., Hunt, A. G., Ewing, R. P., & Sahimi, M. (2013). Tortuosity in porous media: A critical review. *Soil Science Society of America Journal*, 77(5), 1461–1477. https://doi.org/10.2136/sssaj2012.0435
26. Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). *Neural message passing for quantum chemistry*. arXiv. https://doi.org/10.48550/ARXIV.1704.01212
27. Gong, L., & Cheng, Q. (2019). Exploiting edge features for graph neural networks. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 9203–9211. https://doi.org/10.1109/CVPR.2019.00943
28. *Google-research/tuning_playbook*. (2025). [Computer software]. Google Research. https://github.com/google-research/tuning_playbook (Original work published 2023)
29. Gostick, J., Aghighi, M., Hinebaugh, J., Tranter, T., Hoeh, M. A., et al. (2016). OpenPNM: A pore network modeling package. *Computing in Science & Engineering*, 18(4), 60–74. https://doi.org/10.1109/MCSE.2016.49
30. Gostick, J., Khan, Z., Tranter, T., Kok, M., Agnaou, M., Sadeghi, M., & Jervis, R. (2019). Porespy: A python toolkit for quantitative analysis of porous media images. *Journal of Open Source Software*, 4(37), 1296. https://doi.org/10.21105/joss.01296
31. Gostick, J. T. (2017). Versatile and efficient pore network extraction method using marker-based watershed segmentation. *Physical Review E*, 96(2), 023307. https://doi.org/10.1103/PhysRevE.96.023307
32. Guibert, R., Nazarova, M., Horgue, P., Hamon, G., Creux, P., & Debenest, G. (2015). Computational permeability determination from pore-scale imaging: Sample size, mesh and method sensitivities. *Transport in Porous Media*, 107(3), 641–656. https://doi.org/10.1007/s11242-015-0458-0
33. Hall, M. A. (1999). *Correlation-based feature selection for machine learning* [The University of Waikato]. https://researchcommons.waikato.ac.nz/handle/10289/15043
34. Han, Y., & Cundall, P. A. (2011). Lattice Boltzmann modeling of pore-scale fluid flow through idealized porous media. *International Journal for Numerical Methods in Fluids*, 67(11), 1720–1734. https://doi.org/10.1002/fld.2443
35. Hasanov, A. K., Dugan, B., Batzle, M. L., & Prasad, M. (2019). Hydraulic and poroelastic rock properties from oscillating pore pressure experiments. *Journal of Geophysical Research: Solid Earth*, 124(5), 4473–4491. https://doi.org/10.1029/2018JB017276

36. Haugen, H. J., & Bertoldi, S. (2017). Characterization of morphology—3D and porous structure. In *Characterization of Polymeric Biomaterials* (pp. 21–53). Elsevier. https://doi.org/10.1016/B978-0-08-100737-2.00002-9
37. Henderson, N., Brêttas, J. C., & Sacco, W. F. (2010). A three-parameter Kozeny–Carman generalized equation for fractal porous media. *Chemical Engineering Science*, 65(15), 4432–4442. https://doi.org/10.1016/j.ces.2010.04.006
38. Hewamalage, H., Ackermann, K., & Bergmeir, C. (2023). Forecast evaluation for data scientists: Common pitfalls and best practices. *Data Mining and Knowledge Discovery*, 37(2), 788–832. https://doi.org/10.1007/s10618-022-00894-5
39. Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In F. Bach & D. Blei (Eds.), *Proceedings of the 32nd International Conference on Machine Learning* (Vol. 37, pp. 448–456). PMLR. https://proceedings.mlr.press/v37/ioffe15.html
40. Kingma, D. P., & Ba, J. (2014). *Adam: A method for stochastic optimization*. arXiv. https://doi.org/10.48550/ARXIV.1412.6980
41. Kipf, T. N., & Welling, M. (2016). *Semi-supervised classification with graph convolutional networks*. arXiv. https://doi.org/10.48550/ARXIV.1609.02907
42. Kočí, P., Isoz, M., Plachá, M., Arvajová, A., Václavík, M., et al. (2019). 3D reconstruction and pore-scale modeling of coated catalytic filters for automotive exhaust gas aftertreatment. *Catalysis Today*, 320, 165–174. https://doi.org/10.1016/j.cattod.2017.12.025
43. Kozeny, j. (1927) *Ueber Kapillare Leitung des Wassers im Boden. Sitzungsber akad. Wiss. , Wien,* 136(2a), pp 271-306. - References—Scientific research publishing. (n.d.). Retrieved August 16, 2025, from https://www.scirp.org/reference/referencespapers?referenceid=1306904
44. Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., & Viggen, E. M. (2017a). Boundary Conditions for Fluid-Structure Interaction. In *The Lattice Boltzmann Method: Principles and Practice* (pp. 433–491). Springer International Publishing. https://doi.org/10.1007/978-3-319-44649-3
45. Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., & Viggen, E. M. (2017b). MRT and TRTcollision operators. In T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, & E. M. Viggen (Eds.), *The Lattice Boltzmann Method: Principles and Practice* (pp. 407–431). Springer International Publishing. https://doi.org/10.1007/978-3-319-44649-3_10
46. Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., & Viggen, E. M. (2017c). Non-dimensionalisation and choice of simulation parameters. In T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, & E. M. Viggen (Eds.), *The Lattice Boltzmann Method: Principles and Practice* (pp. 265–294). Springer International Publishing. https://doi.org/10.1007/978-3-319-44649-3_7
47. Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., & Viggen, E. M. (2017d). Numerical methods for fluids. In T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, & E. M. Viggen, *The Lattice Boltzmann Method* (pp. 31–58). Springer International Publishing. https://doi.org/10.1007/978-3-319-44649-3_2
48. Krüger, T., Kusumaatmaja, H., Kuzmin, A., Shardt, O., Silva, G., & Viggen, E. M. (2017e). The lattice boltzmann equation. In T. Krüger, H. Kusumaatmaja, A. Kuzmin, O. Shardt, G. Silva, & E. M. Viggen, *The Lattice Boltzmann Method* (pp. 61–104). Springer International Publishing. https://doi.org/10.1007/978-3-319-44649-3_3
49. Lallemand, P., & Luo, L.-S. (2003). Lattice Boltzmann method for moving boundaries. *Journal of Computational Physics*, 184(2), 406–421. https://doi.org/10.1016/S0021-9991(02)00022-0
50. Latt, J., Malaspinas, O., Kontaxakis, D., Parmigiani, A., Lagrava, D., et al. (2021). Palabos: Parallel lattice boltzmann solver. *Computers & Mathematics with Applications*, 81, 334–350. https://doi.org/10.1016/j.camwa.2020.03.022
51. Li, Q., Luo, K. H., Kang, Q. J., He, Y. L., Chen, Q., & Liu, Q. (2016). Lattice Boltzmann methods for multiphase flow and phase-change heat transfer. *Progress in Energy and Combustion Science*, 52, 62–105. https://doi.org/10.1016/j.pecs.2015.10.001
52. Ly, H. B., Monchiet, V., & Grande, D. (2016). Computation of permeability with Fast Fourier Transform from 3-D digital images of porous microstructures. *International Journal of Numerical Methods for Heat & Fluid Flow*, 26(5), 1328–1345. https://doi.org/10.1108/HFF-12-2014-0369
53. Malekzadeh, M., Hajibabaee, P., Heidari, M., Zad, S., Uzuner, O., & Jones, J. H. (2021). Review of graph neural network in text classification. *2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, 0084–0091. https://doi.org/10.1109/UEMCON53757.2021.9666633

54. Marcato, A., Boccardo, G., & Marchisio, D. (2021). A computational workflow to study particle transport and filtration in porous media: Coupling CFD and deep learning. *Chemical Engineering Journal*, 417, 128936. https://doi.org/10.1016/j.cej.2021.128936
55. Marcato, A., Estrada Santos, J., Boccardo, G., Viswanathan, H., Marchisio, D., & Prodanović, M. (2022). Prediction of local concentration fields in porous media with chemical reaction using a multi scale convolutional neural network. *SSRN Electronic Journal*. https://doi.org/10.2139/ssrn.4167602
56. McKay, M. D., Beckman, R. J., & Conover, W. J. (1979). Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 21(2), 239–245. https://doi.org/10.1080/00401706.1979.10489755
57. Mean squared error. (2008). In *The Concise Encyclopedia of Statistics* (pp. 337–339). Springer New York. https://doi.org/10.1007/978-0-387-32833-1_251
58. Meinicke, S., Dubil, K., Wetzel, T., & Dietrich, B. (2020). Characterization of heat transfer in consolidated, highly porous media using a hybrid-scale CFD approach. *International Journal of Heat and Mass Transfer*, 149, 119201. https://doi.org/10.1016/j.ijheatmasstransfer.2019.119201
59. Meyers, J. J., & Liapis, A. I. (1999). Network modeling of the convective flow and diffusion of molecules adsorbing in monoliths and in porous particles packed in a chromatographic column. *Journal of Chromatography A*, 852(1), 3–23. https://doi.org/10.1016/S0021-9673(99)00443-4
60. Mostaghimi, P., Percival, J. R., Pavlidis, D., Ferrier, R. J., Gomes, J. L. M. A., et al. (2015). Anisotropic mesh adaptivity and control volume finite element methods for numerical simulation of multiphase flow in porous media. *Mathematical Geosciences*, 47(4), 417–440. https://doi.org/10.1007/s11004-014-9579-1
61. Nishiyama, N., & Yokoyama, T. (2017). Permeability of porous media: Role of the critical pore size. *Journal of Geophysical Research: Solid Earth*, 122(9), 6955–6971. https://doi.org/10.1002/2016JB013793
62. Niu, Z., Pinfield, V. J., Wu, B., Wang, H., Jiao, K., et al. (2021). Towards the digitalisation of porous energy materials: Evolution of digital approaches for microstructural design. *Energy & Environmental Science*, 14(5), 2549–2576. https://doi.org/10.1039/D1EE00398D
63. Novák, V., Kočí, P., Štěpánek, F., & Marek, M. (2011). Integrated multiscale methodology for virtual prototyping of porous catalysts. *Industrial & Engineering Chemistry Research*, 50(23), 12904–12914. https://doi.org/10.1021/ie2003347
64. Nowak, A., Villar, S., Bandeira, A. S., & Bruna, J. (2017). *Revised note on learning algorithms for quadratic assignment with graph neural networks*. arXiv. https://doi.org/10.48550/ARXIV.1706.07450
65. Ortega, A. (2022). *Introduction to graph signal processing* (1st ed.). Cambridge University Press. https://doi.org/10.1017/9781108552349
66. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., et al. (2019). *Pytorch: An imperative style, high-performance deep learning library*. arXiv. https://doi.org/10.48550/ARXIV.1912.01703
67. Petrasch, J., Meier, F., Friess, H., & Steinfeld, A. (2008). Tomography based determination of permeability, Dupuit–Forchheimer coefficient, and interfacial heat transfer coefficient in reticulate porous ceramics. *International Journal of Heat and Fluid Flow*, 29(1), 315–326. https://doi.org/10.1016/j.ijheatfluidflow.2007.09.001
68. Rabbani, A., Babaei, M., Shams, R., Wang, Y. D., & Chung, T. (2020). DeePore: A deep learning workflow for rapid and comprehensive characterization of porous materials. *Advances in Water Resources*, 146, 103787. https://doi.org/10.1016/j.advwatres.2020.103787
69. Richter, M. L., Byttner, W., Krumnack, U., Wiedenroth, A., Schallner, L., & Shenk, J. (2021). (Input) Size Matters for CNN Classifiers. In I. Farkaš, P. Masulli, S. Otte, & S. Wermter (Eds.), *Artificial neural networks and machine learning – icann 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part I* (Vol. 12891, pp. 133–144). Springer International Publishing. https://doi.org/10.1007/978-3-030-86362-3
70. Santos, J. E., Yin, Y., Jo, H., Pan, W., Kang, Q., et al. (2021). Computationally efficient multiscale neural networks applied to fluid flow in complex 3d porous media. *Transport in Porous Media*, 140(1), 241–272. https://doi.org/10.1007/s11242-021-01617-y
71. Schulz, R., Ray, N., Zech, S., Rupp, A., & Knabner, P. (2019). Beyond Kozeny–Carman: Predicting the permeability in porous media. *Transport in Porous Media*, 130(2), 487–512. https://doi.org/10.1007/s11242-019-01321-y
72. Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1), 60. https://doi.org/10.1186/s40537-019-0197-0

73. Singh, A., Regenauer-Lieb, K., Walsh, S. D. C., Armstrong, R. T., Van Griethuysen, J. J. M., & Mostaghimi, P. (2020). On representative elementary volumes of grayscale micro-CT images of porous media. *Geophysical Research Letters*, 47(15), e2020GL088594. https://doi.org/10.1029/2020GL088594
74. Tarokh, A., Mohamad, A. A., & Jiang, L. (2013). Simulation of conjugate heat transfer using the lattice boltzmann method. *Numerical Heat Transfer, Part A: Applications*, 63(3), 159–178. https://doi.org/10.1080/10407782.2012.725009
75. Wagner, A., Eggenweiler, E., Weinhardt, F., Trivedi, Z., Krach, D., et al. (2021). Permeability estimation of regular porous structures: A benchmark for comparison of methods. *Transport in Porous Media*, 138(1), 1–23. https://doi.org/10.1007/s11242-021-01586-2
76. Wang, H., Yin, Y., Hui, X. Y., Bai, J. Q., & Qu, Z. G. (2020). Prediction of effective diffusivity of porous media using deep learning method based on sample structure information self-amplification. *Energy and AI, 2,* 100035. https://doi.org/10.1016/j.egyai.2020.100035
77. Wang, M., & Pan, N. (2007). Numerical analyses of effective dielectric constant of multiphase microporous media. *Journal of Applied Physics*, 101(11), 114102. https://doi.org/10.1063/1.2743738
78. Wang, M., Wang, J., Pan, N., Chen, S., & He, J. (2007). Three-dimensional effect on the effective thermal conductivity of porous media. *Journal of Physics D: Applied Physics*, 40(1), 260–265. https://doi.org/10.1088/0022-3727/40/1/024
79. Wu, H., Fang, W.-Z., Kang, Q., Tao, W.-Q., & Qiao, R. (2019). Predicting effective diffusivity of porous media from images by deep learning. *Scientific Reports*, 9(1), 20387. https://doi.org/10.1038/s41598-019-56309-x
80. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1), 4–24. https://doi.org/10.1109/TNNLS.2020.2978386
81. Xiong, Q., Baychev, T. G., & Jivkov, A. P. (2016). Review of pore network modelling of porous media: Experimental characterisations, network constructions and applications to reactive transport. *Journal of Contaminant Hydrology*, 192, 101–117. https://doi.org/10.1016/j.jconhyd.2016.07.002
82. Yagis, E., Atnafu, S. W., García Seco De Herrera, A., Marzi, C., Scheda, R., et al. (2021). Effect of data leakage in brain MRI classification using 2D convolutional neural networks. *Scientific Reports*, 11(1), 22544. https://doi.org/10.1038/s41598-021-01681-w
83. Yasuda, T., Ookawara, S., Yoshikawa, S., & Matsumoto, H. (2021). Machine learning and data-driven characterization framework for porous materials: Permeability prediction and channeling defect detection. *Chemical Engineering Journal*, 420, 130069. https://doi.org/10.1016/j.cej.2021.130069
84. Ye, L., Lv, W., Zhang, K. H. L., Wang, X., Yan, P., et al. (2015). A new insight into the oxygen diffusion in porous cathodes of lithium-air batteries. *Energy*, 83, 669–673. https://doi.org/10.1016/j.energy.2015.02.072
85. Zhang, Y., Yang, Z., Wang, F., & Zhang, X. (2021). Comparison of soil tortuosity calculated by different methods. *Geoderma*, 402, 115358. https://doi.org/10.1016/j.geoderma.2021.115358
86. Zhao, Q., Han, X., Guo, R., & Chen, C. (2025). *A computationally efficient hybrid neural network architecture for porous media: Integrating convolutional and graph neural networks for improved property predictions* (No. arXiv:2311.06418). arXiv. https://doi.org/10.48550/arXiv.2311.06418