Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

Reconstruction and Generation of Porous Metamaterial Units via Variational Graph Autoencoder and Large Language Model

Kiarash Naghavi Khanghah, Zihan Wang, Hongyi Xu*

School of Mechanical, Aerospace, and Manufacturing Engineering
University of Connecticut, Storrs, CT 06269
* Email: hongyi.3.xu@uconn.edu

ABSTRACT

In this paper, we propose and compare two novel deep generative model-based approaches for the design representation, reconstruction, and generation of porous metamaterials characterized by complex and fully connected solid and pore networks. A highly diverse porous metamaterial database is curated, with each sample represented by solid and pore phase graphs and a voxel image. All metamaterial samples adhere to the requirement of complete connectivity in both pore and solid phases. The first approach employs a Dual Decoder Variational Graph Autoencoder to generate both solid phase and pore phase graphs. The second approach employs a Variational Graph Autoencoder for reconstructing/generating the nodes in the solid phase and pore phase graphs and a Transformer-based Large Language Model (LLM) for reconstructing/generating the connections, i.e., the edges among the nodes. A comparative study was conducted, and we found that both approaches achieved high accuracy in reconstructing node features, while the LLM exhibited superior performance in reconstructing edge features. Reconstruction accuracy is also validated by voxel-to-voxel comparison between the reconstructions and the original images in the test set. Additionally, discussions on the advantages and limitations of using LLMs in metamaterial design generation, along with the rationale behind their utilization, are provided.

Keywords: Porous Metamaterial; Graph Representation; Graph Neural Network; Large Language Model; Variational Graph Autoencoder.

1. INTRODUCTION

Various metamaterials have been developed to achieve exceptional mechanical properties, catering to diverse applications [1-12]. Their extraordinary mechanical characteristics are attributed to their distinctive topological features. The design of porous metamaterials suitable for applications involving fluid-filled conditions [13-15] has been relatively overlooked despite a substantial body of research in the field of metamaterial research. The use of conventional techniques like parametric design and analytical modeling typically restricts the design freedom of metamaterials with fluid-filled porous structures [1-5]. Therefore, new approaches enabling the freeform design of porous metamaterials, which satisfy the criterion of complete connectivity in both pore and solid phases [16], must be established. “Complete connectivity” means that there are no isolated solid parts or pores within the structure. However, detecting and repairing such disconnection is challenging, and methods such as texture synthesis [16] and the virtual temperature method [17] either, in some cases, cannot guarantee complete connectivity or are computationally expensive. Moreover, earlier studies on the design of porous metamaterial units either restrict design possibilities to simple structures like lattices, simplifying connectivity verification or fail to ensure complete connectivity in both solid and pore components [18-22].

A promising solution to this challenge is to employ graph representation for designing metamaterials [23-25] and microstructures [26]. Derived from graph theory, the graph representation-based methods are computationally efficient in detecting disconnections and isolated parts and creating fully connected structures. Graph-based methods have been widely employed in materials science [23, 26, 27], chemistry [28], and structure design for mechanical properties [29-33]. In our previous work [34], we showcased the efficiency of graph-based representation in creating porous metamaterial structures with *complete*

connectivity in both pore and solid phases. The graph representation of porous metamaterials introduced in that study *serves as the basis for the study in this paper*.

Graph representation enables the application of graph neural networks (GNNs)[23, 27, 35-37] and large language models (LLMs), which require structured data—a feature inherently provided by graph structures. Each graph contains nodal end connection information $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$, and the design and fabrication of porous metamaterials can be achieved by configuring these nodes and their connections (edges) [23]. A major challenge associated with GNN is to predict the edges for a given set of nodes accurately. Zhang [38] categorized the edge prediction approaches into two main categories: subgraph-based methods and node-based methods. The subgraph-based methods, such as learning from Subgraphs, Embeddings, and Attributes for Link prediction (SEAL) frameworks [39], necessitate a partially connected graph as the starting point to generate connections among the remaining nodes. On the other hand, node-based methods, such as the Variational Graph Autoencoders (VGAE) [40, 41], learn the underlying distribution of graphs from a given dataset, and generate new graphs following the same distribution. However, VGAE face challenges in capturing complex and high-dimensional graph structures due to the information loss in the latent space, Additionally, they may struggle to capture long-range dependencies or global structural features within the graph due to their inability to account for the relative positions and associations between them [38].

In contrast, Transformer-based LLMs are powerful in predicting edges in graphs because they can comprehend long-range dependencies and consider various relationships before establishing connections [42, 43]. LLMs have been utilized in various graph-based problems [44-49]. However, despite their promising abilities, such as detecting the relation between dataset and zero-shot learning [50-52], their successful application in structural design, including the design of metamaterial structures, is still lacking. It is important to note that LLMs work best for cases where the data can be represented in a sequential or structured format (such as graphs and natural languages) [53]. Hence, we propose to leverage LLM in designing graph-represented microstructures in this study.

Despite these differences, GNN and LLM can be used entangled with each other. LLM can be used as an enhancer of GNN[42, 44] by capturing node embeddings containing long-range relations with other nodes; GNN can create graph embeddings as fine-tuning inputs for LMM to improve LLM efficiency [42, 44, 53, 54]. Additionally, these two models can operate in parallel [42, 44]. This highlights the potential of LLMs in engineering design problems where structured graph data is involved.

The remainder of this paper is organized as follows. Section 2 introduces the proposed methodologies based on VGAE and LLM, for design representation, reconstruction, and generation of porous metamaterials. Section 3 presents a case study to quantitatively assess the effectiveness of the proposed methodologies. Section 4 discusses the advantages of employing LLM in porous metamaterial design problems. Section 5 concludes this work.

2. METHODOLOGY: GENERATIVE MODEL-BASED APPROACHES FOR GENERATING POROUS METAMATERIAL STRUCTURES

Two new approaches are proposed for design representation, reconstruction, and generation of porous metamaterials. As discussed in Section 2.1, porous metamaterial samples have been generated based on a solid phase graph and a pore phase graph in order to guarantee complete connectivity in both phases. The first approach, presented in Sections 2.2 and 2.3, employs a Dual Decoder Variational Graph Autoencoder (DVGAE) for predicting both nodes and edges of the graphs that represent the solid and pore phases in the porous structure. The second approach, presented in Section 2.4, uses VGAE as the node generator and a fine-tuned LLM as the edge generator. A comparative study of the two approaches will be presented in Section 3. Section 2.1 introduces the approach for graph-based representation and generation of diverse training samples, which are porous metamaterial units with complete connectivity in both pore and solid phases.

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

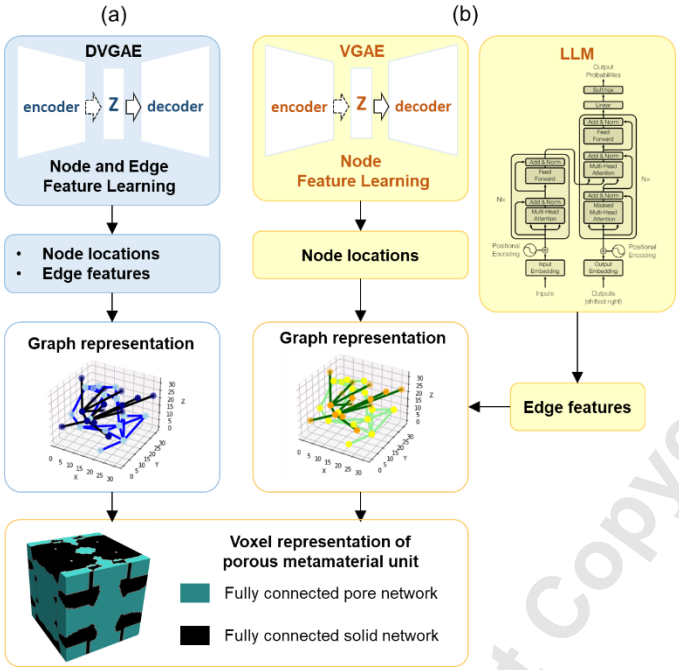


Figure 1: Proposed approaches for generating complex porous metamaterial designs. (a) Dual Decoder Variational Graph Autoencoder (DVGAE) for generating both nodes and edges and (b) A hybrid approach that integrates VGAE and LLM (LLM model architecture presented by Vaswani et al. [43]).

2.1 TRAINING DATASET: POROUS METAMATERIAL SAMPLES WITH COMPLETE CONNECTIVITY IN BOTH SOLD AND PORE PHASES

We proposed a graph-based approach for generating complex porous microstructures with complete connectivity in both solid and pore phases (refer to [34] for details). This approach involves constructing the porous metamaterial unit from two "interwoven" graphs that represent the solid phase and the pore phase, respectively. As shown in Figure 2, this approach consists of the following steps.

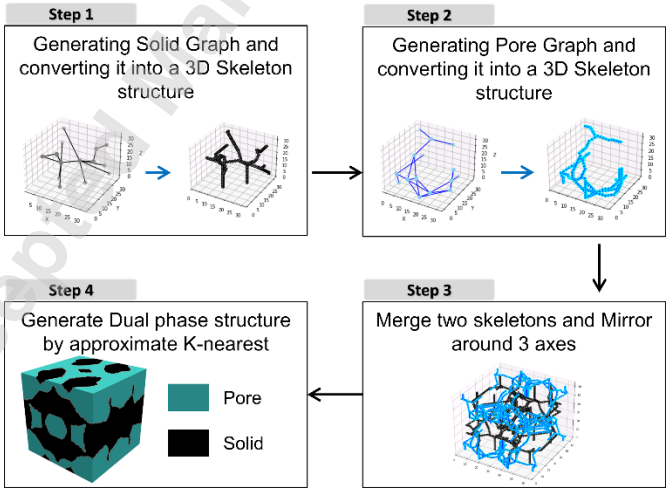


Figure 2: The proposed approach for generating complex porous metamaterial unit samples with complete connectivity in both solid and pore phases-demonstrating using one of the most complex samples.

In the first step, a graph representing the skeleton of the solid phase is created ("solid phase graph"). The nodes of the solid graph are randomly selected, and then connected based on a distance-based logic.

This logic connects neighboring nodes within a specific distance until all nodes are interconnected, in the case of isolated node clusters, long-range connections will be established to achieve a single connected graph. The nodes and edges of the graph are then mapped onto a voxel grid to create the voxel skeleton of the solid phase.

The second step is to create the "pore phase graph" in a similar way. When creating the voxel skeleton of the pore phase, if the path between a pair of nodes is blocked by a voxel in the solid skeleton, a rerouting strategy based on the Manhattan method [55] is conducted to bypass the blocking voxel.

In the third step, the two skeletons are merged into the same space (1/8 of the entire 3D space), and mirror operations are conducted to create a symmetric structure.

In the last step, the remaining unlabeled voxels in the 3D image are assigned to either the solid phase or the pore phase by the approximate K-nearest neighbor-based clustering using a K-Dimensional tree (K-D tree) [56], after merging the two skeletons into one voxel image and mirroring along all three axes to create a symmetric metamaterial unit structure.

Both solid and pore phases in the created metamaterial design are *inherently fully connected*, which is a major advantage of this approach. It does not need any additional post-processing to remove "enclosed voids" or "disconnected materials". In this study, by controlling the distance-based connection radius, two sets of graph samples with different levels of complexity have been created. By randomizing the locations of the input nodes, highly diverse metamaterial unit datasets are generated (Figure 3). This database includes three datasets (Table 1). Dataset 1 has 130,961 samples of simple graphs with a fixed number of nodes, which is 15 in this case. Dataset 2 has 300,000 samples of complex graphs with a fixed number of nodes, which is 15 in this case too. Dataset 3 has 500,000 samples of complex graphs with a varying number of nodes, ranging from 12 to 60 nodes. The "complexity" of the graph is measured by the metric of average eccentricity [57]. Dataset 1 (simple graphs) has an average eccentricity of 2.78, while Datasets 2 and 3 (complex graphs) have average eccentricity values of 5.75 and 8.36, respectively.

Table 1: Number of samples, number of nodes for constructing graphs, and graph complexity of the samples in the three datasets

Case	Number of samples	Number of nodes for constructing graphs	Graph Complexity
Dataset 1	130,961	15	2.78
Dataset 2	300,000	15	5.75
Dataset 3	500,000	12-60	8.36

Both DVGAE and Hybrid models will be trained and tested on all 3 datasets. For Dataset 1, 129,961 samples are used for training and 1,000 samples are used for testing; For Dataset 2, 295,000 samples are used for training and 5,000 samples are used for testing, and for Dataset 3, 495,000 samples are used for training and 5,000 samples are used for testing. This dataset was used exclusively for the training and validation of DVGAE and Hybrid models, while the rules used to create it remain unknown to both VGAE and LLM models. The ingenuity of this work lies in its ability to generate structures without knowing the rules which means these models can capture the generation rules of any dataset and regenerate the structure.

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

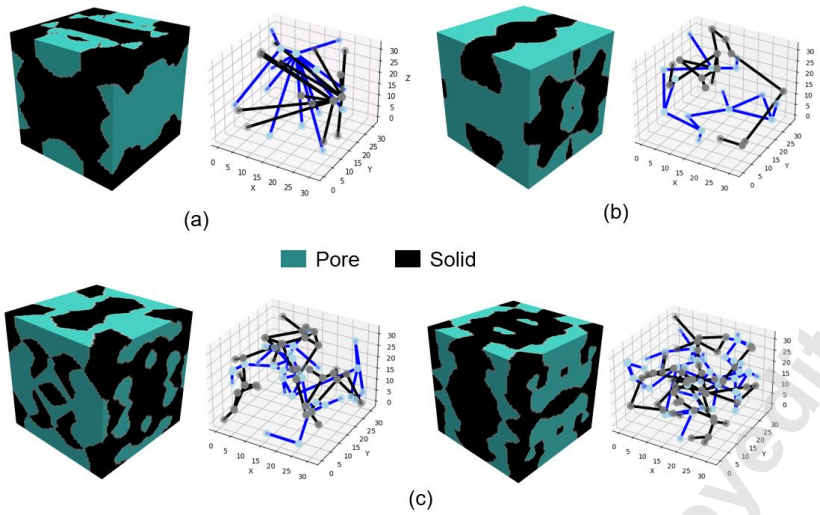


Figure 3: Diversity of the training samples: several examples of metamaterial units, the voxel image, and the graph representation of 1/8 of the cube, in the created database. (a) A sample from Dataset 1 – simple graphs. (b) A sample from Dataset 2 – complex graph. (c) Samples from Dataset 3 – complex graphs and varying number of nodes.

2.2 Dual Decoder Variational Graph Autoencoder (DVGAE) for Both Node and Edge Feature Learning

Graph autoencoder (GAE) and its variations (e.g., VGAE [58], adversarial regularized graph autoencoder [59], deep attention embedding graph autoencoder [60], etc.) to learn latent representations of graphs using an autoencoder framework have been widely used to generate new graphs. DVGAE [61] is employed in this work to generate node attributes and graph structures (edges) simultaneously. Furthermore, by employing a variational framework, our aim is to more accurately capture the inherent distribution of graph data, thereby improving the model's capacity to generate novel graphs. A DVGAE comprises the following four major components:

(1) Graph Convolutional Encoder, $q(\mathbf{z}|\mathbf{X},\mathbf{A})$, which can be expressed as:

$$q(\mathbf{z}|\mathbf{X},\mathbf{A}) = \Pi_{i=1}^{N} q(\mathbf{z}_i|\mathbf{X},\mathbf{A}) \quad (1)$$

$$q(\mathbf{z}_i|\mathbf{X},\mathbf{A}) = N(\mathbf{z}_i|\boldsymbol{\mu}_i, \mathrm{diag}(\boldsymbol{\sigma}_i{}^2)) \quad (2)$$

where we define an undirected, unweighted graph $G = (v, \varepsilon)$ with $N = |v|$ nodes. $\mathbf{X}$ is the node features matrix of the graph G, $\mathbf{A}$ is the adjacency matrix of the graph G, and $\mathbf{D}$ is the normalized degree matrix of G. $\mathbf{z}_i$ represents a latent variable, and the latent vector $\mathbf{z}$ is an $N \times F$ matrix, where F is the dimension of the latent vectors to which each node is mapped. $\mathbf{X}$ represents the node features matrix with a dimension of $N \times D$. The mean ($\boldsymbol{\mu}_i$) and variance ($\boldsymbol{\sigma}_i{}^2$) of the latent variables for each node are computed using two GCN layers: one for the means ($\boldsymbol{\mu} = \mathrm{GCN}_{\boldsymbol{\mu}}(\mathbf{X},\mathbf{A})$) and another for the log variance ($\log\boldsymbol{\sigma} = \mathrm{GCN}_{\boldsymbol{\sigma}}(\mathbf{X},\mathbf{A})$). During the GCN operation, for each graph, given the node feature matrix $\mathbf{X}$ and the edge feature matrix $\mathbf{A}$, we then have $\mathbf{H} = \mathbf{A}'\mathbf{X}\mathbf{W}$, where $\mathbf{W}$ is the trainable weight matrix, and $\mathbf{A}' = \mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{\frac{1}{2}}$.

(2) Inner Product Decoder, which is constructed with fully connected layers, takes the latent vector $\mathbf{Z}$ as input to reconstruct the original graph G. The underlying assumption of this decoding strategy is that if two nodes are similar in the latent space (i.e., their latent vectors are close to each other), they are more likely to be connected in the graph. The reconstructed adjacency matrix $\widetilde{\mathbf{A}}$ is reconstructed as:

$$\widetilde{\mathbf{A}} = \sigma(\mathbf{Z}\mathbf{Z}^{\mathbf{T}}) \quad (3)$$

where $\sigma(\cdot)$ is the logistic sigmoid function, which ensures that the output values are in the range (0,1), interpretable as probabilities.

(3) Graph Convolutional Decoder, which consists of a graph convolutional layer followed by a node-wise softmax operation to reconstruct node features $\widetilde{\mathbf{X}}$. The graph convolutional decoder is defined as:

$$\widetilde{\mathbf{X}} = f(\mathbf{Z}, \mathbf{A}) = \underline{\mathbf{A}}\, ReLU\big(\underline{\mathbf{A}}\mathbf{Z}\mathbf{W}^{(0)}\big)\mathbf{W}^{(1)} \tag{4}$$

where $\mathbf{A}$ is the adjacency matrix. $\mathbf{Z}$ represents the latent representation obtained from the encoder. $\mathrm{ReLU}(\cdot) = (0; \cdot)$ is a nonlinear activation function. $\mathbf{W}$ represents the trainable weight matrix. The structure of the node and edge features is utilized through the entire encoding-decoding process, owing to the usage of the graph convolutional layers in both the encoder and decoder.

(4) Loss Function, which consists of two parts: the reconstruction loss and the Kullback-Leibler divergence loss. The reconstruction loss comes from both the inner product decoder and the graph convolutional decoder. The inner product decoder reconstructs the adjacency matrix, and the associated loss function is defined as:

$$\mathcal{L}_{adj} = \mathrm{E}_{q(\mathbf{Z}|\mathbf{X},\mathbf{A})}[\log p(\mathbf{A}|\mathbf{Z})] \tag{5}$$

where $q(\mathbf{Z}|\mathbf{X},\mathbf{A})$ is the posterior inference, which can be recognized as performing posterior inference over all the data points in the dataset, where:

$$p(\mathbf{A}|\mathbf{Z}) = \Pi_{i=1}^{N}\Pi_{j=1}^{N} p\big(\mathrm{A}_{ij}\big|\mathbf{z}_i, \mathbf{z}_j\big) \tag{6}$$

$$p\big(\mathrm{A}_{ij} = 1\big|\mathbf{z}_i, \mathbf{z}_j\big) = \boldsymbol{\sigma}(\mathbf{z}_i{}^{\mathrm{T}}\mathbf{z}_j) \tag{7}$$

The graph convolutional decoder reconstructs the node feature matrix, and the associated loss function is defined as:

$$\mathcal{L}_X = \frac{1}{2}\left|\left|\mathbf{X} - \widetilde{\mathbf{X}}\right|\right|^2 \tag{8}$$

The VGAE model is trained to optimize the variational lower bound L:

$$\mathcal{L}_{VGAE} = \lambda_1\mathcal{L}_{adj} + \lambda_2\mathcal{L}_X + \lambda_3\mathcal{L}_{KL} = \mathrm{E}_{q(\mathbf{Z}|\mathbf{X},\mathbf{A})}[\log p(\mathbf{A}|\mathbf{z})] + \frac{1}{2}\left|\left|\mathbf{X} - \widetilde{\mathbf{X}}\right|\right|^2 - \mathrm{KL}[q(\mathbf{z}|\mathbf{X},\mathbf{A})||p(\mathbf{z})] \tag{9}$$

where $\mathrm{KL}[q(\cdot)||p(\cdot)]$ is the Kullback-Leibler divergence between $q(\cdot)$ and $p(\cdot)$. We use Gaussian prior $p(\mathbf{z}) = \Pi_i p(\mathbf{z}_i) = \Pi_i N(\mathbf{z}_i|0, \mathbf{I})$. To optimize the parameters of the Gaussian distribution, we perform mini-batch gradient descent and leverage the reparameterization trick [62]. λ_1, λ_2 and λ_3 are coefficients to balance different loss terms to achieve better accuracy. In this work, we use $\lambda_1 = \lambda_2 = \lambda_3 = 1$.

To show the advantages of using the DVGAE model, we compared it to a baseline model that is constructed using dense layers in a dual-decoder VAE (DVAE) frame instead of using graph layers.

2.3 DVGAE Model Training

For the metamaterial samples in our dataset, both solid and pore phases are represented by a 15-node graph. The node feature matrix $\mathbf{X}$ contains the coordinates ($\mathbf{X}_x$, $\mathbf{X}_y$, $\mathbf{X}_z$) of each node, and the edge feature matrix $\mathbf{A}$ representing the connection between nodes. For model training, we use the PyTorch Geometric library. The models are trained on Nvidia RTX8000. Adam is used as the optimizer for parameter optimization. The training and test datasets are introduced in Section 2.1.

2.4 Hybrid approach: VGAE for node generation and LLM for edge generation.

The proposed hybrid approach utilizes VGAE to generate the node features of the graph and establishes connections among the nodes using LLM. The VGAE for node generation follows the same methodology as presented in Section 2.1.

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

The remarkable capabilities of LLMs in processing structured data have inspired the utilization of LLMs in graph-based problems [44]. As described in our previous work [34] and Section 2.1, the porous structure is represented by two graphs, one for the solid phase and the other for the pore phase. In each graph, the nodes represent the joints in the solid/pore networks, and the edges represent the conduits/connections between neighbor joints. The reason behind employing LLM lies in the conceptualization of the solid phase and pore phase graphs as sequential data, where node connections are determined by distances. Therefore, sequence-to-sequence (seq2seq) learning [63, 64] is employed to get nodes' positional information as input and predict the nodes' connections as the output.

In this paper, the byt5 model has been utilized as the LLM for edge prediction [65, 66]. ByT5 is a pre-trained LLM, which offers better generalizability compared to task-specific transformer models, enabling fine-tuning with a smaller number of graph data [67]. ByT5, similar to other variants of the Text-to-Text Transfer Transformer (T5) model, has an Encoder-Decoder architecture [68]. The encoder-decoder framework is well-suited for seq2seq tasks due to its capability to maintain effective attention on both source and target sequences [69]. By having an attention mechanism [43] in both decoder and encoder, it can detect hard-to-detect dependencies, which, given the fact that the connection of a graph could be a difficult task, makes it crucial. Furthermore, Wang et al. [50] demonstrated that models employing an encoder-decoder structure, when fine-tuned on multiple tasks, exhibit the highest zero-shot capabilities. This implies that while the database has been created with a fixed number of nodes, the LLM model possesses the flexibility to accept an arbitrary number of nodes as input and generate their connections, a capability often termed as zero-shot learning [48, 70].

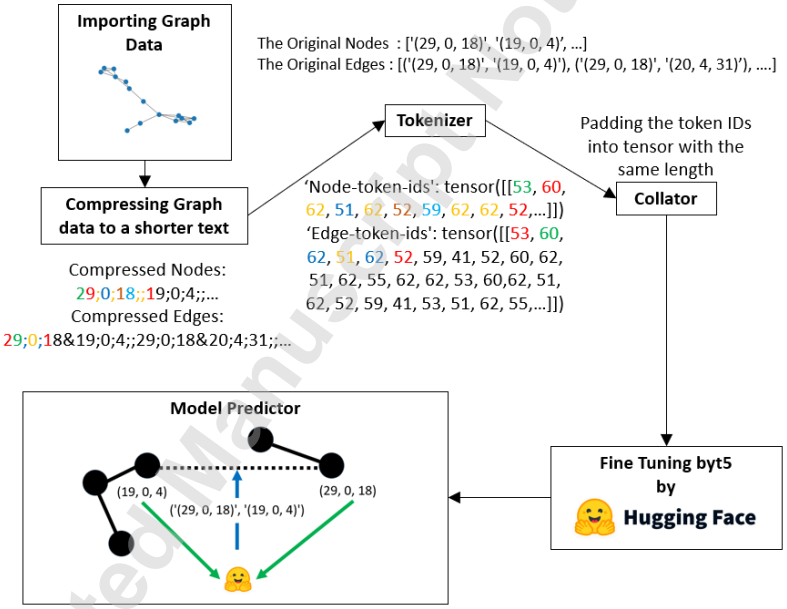


Figure 4: The approach for fine-tuning the byt5-base model for the task of predicting node connections in graphs that represent the solid and pore phases in porous metamaterials (The Hugging Face logo is provided by the Hugging Face: https://huggingface.co/brand).

Among the different variations of T5, ByT5 stands out as a byte-level transformer model, also known as a token-free model. This approach interprets sequences as UTF-8 bytes, and according to Xue et al. [65], it performs particularly well in tasks involving numerical reasoning and those at the byte level. This model could be advantageous for training on graph data containing positional information, as it necessitates numerical reasoning to establish node connections. For byt5 to be able to do accurate predictions, it needs to be trained on graph data as the downstream task. Furthermore, employing a dataset with varying numbers of nodes in training the byt5 model enhances both accuracy and generalizability in zero-shot learning scenarios [71, 72]. In the proposed approach, as depicted in Figure 4, the following steps are undertaken.

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

In the first step of finetuning the LLM model, the graph data as a downstream task will be imported. Since the LLM requires text input data, using a standard process the graph will be turned into text where compression can be applied to enhance performance while reducing memory and computational complexity[66]. This compression is necessary to increase performance when leveraging attention mechanisms, which typically have a complexity of $O(n^2)$ [43, 66, 73].

The compressed data will be tokenized using the byT5 special tokenizer, converted into tensors of the same length via a collator, and then the byt5-base model will be fine-tuned using the seq2seq trainer, all from the Hugging Face library [66, 74, 75]. It is notable that, as demonstrated in Figure 4, the compressed input sequence follows the format "node1;;node2;;…", while the compressed output sequence adopts the format "node j@node k;; node i@node m;;...". Additionally, each node is characterized by three dimensions "X_x; X_y; X_z", which also needs to be within a reasonable range.

During the fine-tuning process, all model parameters are retrained on the new graph data (in this study, they are the metamaterial structures in graph representation introduced in Section 2.1). The objective of the fine-tuned model is to predict node connectivity by feeding nodal information to the LLM, facilitating the creation of fully connected complex porous materials. The remarkable aspect of this model is its capability to predict edge connections without prior knowledge of a partially connected graph. Only node features are needed as the input for edge generation.

The LLM predicts all the connections of the given nodes in a single computational process. Also, it should be noted that the LLM also learns the graph generation, extraction, and saving rules. In this study, the NetworkX library [76] was used to archive our training dataset and the LLM, due to its “black box” [77] nature seems to learn the underlying rules behind graph generation, extraction, and archiving. Consequently, the current fine-tuned model is not invariant to the ordering of input nodes, even though no specific ordering was enforced in the training data. Further investigation into fine-tuning an LLM that is invariant to ordering is needed in our future works. Figure 5 presents a comparison between the graph predicted by an LLM, illustrated in green, and the original graph derived from the test dataset, illustrated in blue, with respect to the connections among the nodes. This comparison reveals a high accuracy in edge prediction when utilizing the LLM can be achieved.

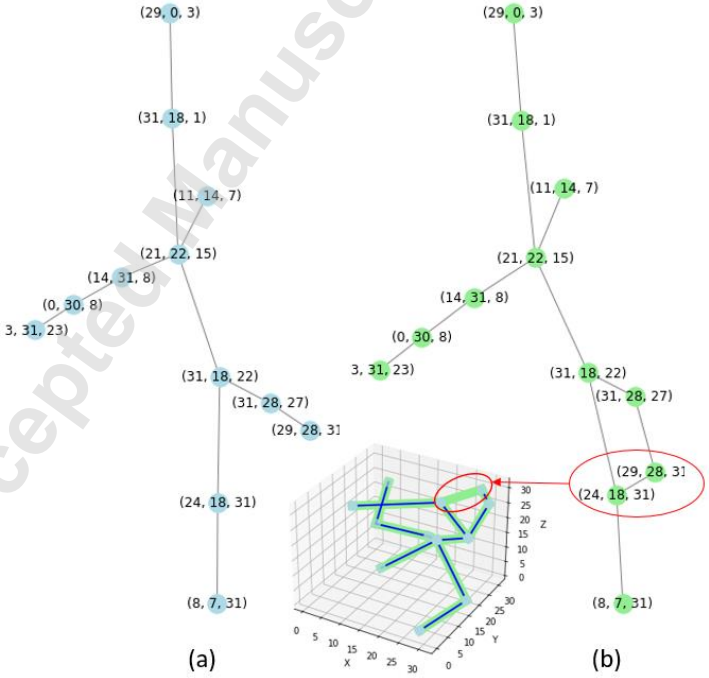


Figure 5: Demonstrating Graph Prediction: (a) A ground truth graph from the test set and (b) the predicted graph by LLM.

One inherent capability of LLMs is zero-shot learning, allowing the model to make accurate predictions even without prior exposure to a dataset [50-52]. This capability offers two advantages in porous metamaterial modeling. Firstly, although trained on solid phase graphs, the model can accurately predict pore phase graphs, which are generated following the same logic as the solid phase graph. Secondly, in the case that the training dataset comprises a fixed number of nodes, the LLM model can predict outputs even with an arbitrary number of input nodes. The resulting connections adhere to the same logic as those generated in Section 2.1, enhancing the flexibility in design generation by adding different number of nodes during the design. Consequently, an LLM-based generative model can be employed for metamaterial structure generation.

To demonstrate the accuracy and predictive capabilities of fine-tuned models and ensure that the appropriate model has been selected for fine-tuning, two additional models are explored and compared. These include another byt5-base model with a prompt including an input node, an example node, and an edge to generate the link without fine-tuning, as well as another encoder-decoder model, BART [78], fine-tuned with the same dataset. Detailed information regarding the new LLM model and its comparison with the proposed Byt5 model can be found in the Appendix.

3. RESULTS OF COMPARATIVE STUDIES

3.1 Reconstruction Accuracy of the Baseline VAE Model

The accuracy of the DVAE model on three datasets is presented in the tables below. Reconstruction of node locations is a relatively simple task, while the major challenge is to reconstruct the edges accurately, which is the focus of the following comparative studies.

The accuracy of the DVAE model is evaluated based on two criteria: the accuracy of the reconstructed node features map and the reconstructed adjacency matrix. The accuracies of reconstructing node features map $\widetilde{\mathbf{X}}$ and adjacency matrix $\widetilde{\mathbf{A}}$ are evaluated by calculating the coefficient of determination (R^2) values [79, 80], which measures the degree of agreement between the original and reconstructed samples:

$$R^2 = 1 - \frac{\sum(\mathbf{Y}_i - \widetilde{\mathbf{Y}}_i)^2}{\sum(\mathbf{Y}_i - \overline{\mathbf{Y}}_i)^2} \tag{10}$$

where $\mathbf{Y}_i$ represents the true response of the i^{th} sample, $\widetilde{\mathbf{Y}}_i$ represents the predicted response of the i^{th} sample, and n_{sample} represents the total number of sample points. $\overline{\mathbf{Y}}_i$ is the averaged value of $\mathbf{Y}_{i(true)}$ and $\overline{\mathbf{Y}}_i = \frac{1}{n_{sample}} \sum \mathbf{Y}_i$. The accuracy of edge prediction is measured by assessing each possible pair of nodes within the graph and determining if there is a link (edge) between them. The accuracy of link predictions is then measured as the percentage of pairs that the model predicted correctly which matches the actual presence of an edge in the graph from the test dataset. The results reveal that DVGAE successfully reconstructs $\widetilde{\mathbf{X}}$**,** but totally fails to reconstruct $\widetilde{\mathbf{A}}$.

Table 2: Reconstruction Accuracies of both solid phases and pore phases with simple graphs (Dataset 1).

Phase		$\widetilde{\mathbf{X}}$	$\widetilde{\mathbf{A}}$	**Link Prediction Accuracy**
Solid	Training	0.999	-424865	0.35%
	Test	0.999	-427562	0.11%
Pore	Training	0.999	-412085	0.42%
	Test	0.999	-425529	0.21%

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

Table 3: Reconstruction Accuracies of both solid phases and pore phases with complex graphs (Dataset 2).

Phase		$\widetilde{\mathbf{X}}$	$\widetilde{\mathbf{A}}$	**Link Prediction Accuracy**
Solid	Training	0.999	-427562	0.13%
	Test	0.999	-431185	0.09%
Pore	Training	0.999	-430215	0.11%
	Test	0.999	-436521	0.10%

Table 4: Reconstruction Accuracies of both solid phases and pore phases with complex graphs (Dataset 3).

Phase		$\widetilde{\mathbf{X}}$	$\widetilde{\mathbf{A}}$	**Link Prediction Accuracy**
Solid	Training	0.999	-421526	0.24%
	Test	0.999	-430425	0.14%
Pore	Training	0.999	-429850	0.18%
	Test	0.999	-430114	0.13%

3.2 Reconstruction accuracy of the DGVAE Model

The accuracy of the DVGAE model is also evaluated based on the accuracy of the reconstructed node features map and the accuracy of the reconstructed adjacency matrix. The reconstruction accuracies of both solid phase and pore phase graphs for all three datasets are shown in Table 5-7.

The results reveal that DVGAE successfully reconstructs $\widetilde{\mathbf{X}}$, but fails to reconstruct $\widetilde{\mathbf{A}}$ accurately. Compared with the DVAE model, the prediction accuracy of $\widetilde{\mathbf{A}}$ is slightly higher. This is because the dense layers do not inherently capture the graph structure and the relationships between nodes and edges; they treat the input data as a flat, unstructured collection of features. The failure of reconstruction of $\widetilde{\mathbf{A}}$ could be attributed to the DVGAE's limited ability to fully comprehend the complex structure of the graph. The objective of the inner product decoder for edge reconstruction is to establish connections between nodes while adhering to the logic used in generating the training data. The logic does not solely depend on neighbor distance to link nodes. As we discussed in Section 2.1, long-range connections are also established between isolated node clusters. Therefore, capturing this intricate connection rule presents a challenge for DVGAE.

Table 5: Reconstruction Accuracies of both solid phases and pore phases with simple graphs (Dataset 1).

Phase		$\widetilde{\mathbf{X}}$	$\widetilde{\mathbf{A}}$	**Link Prediction Accuracy**
Solid	Training	0.999	-1.924	7.32%
	Test	0.999	-8.152	2.58%
Pore	Training	0.999	-1.936	7.22%
	Test	0.999	-8.235	2.74%

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

Table 6: Reconstruction Accuracies of both solid phases and pore phases with complex graphs (Dataset 2).

Phase		$\tilde{\mathbf{X}}$	$\tilde{\mathbf{A}}$	**Link Prediction Accuracy**
Solid	Training	0.999	-1.243	12.76%
	Test	0.999	-6.461	6.33%
Pore	Training	0.999	-1.276	12.53%
	Test	0.999	-6.319	6.71%

Table 7: Reconstruction Accuracies of both solid phases and pore phases with complex graphs (Dataset 3).

Phase		$\tilde{\mathbf{X}}$	$\tilde{\mathbf{A}}$	**Link Prediction Accuracy**
Solid	Training	0.999	-2.014	8.77%
	Test	0.999	-7.364	3.48%
Pore	Training	0.999	-2.078	8.68%
	Test	0.999	-7.268	3.66%

3.3 Reconstruction Accuracy of the LLM Model

Compared with DGVAE, byt5 exhibits better prediction accuracies in edge prediction. The edge prediction average accuracies of both solid and pore phase graphs are shown in Table 8. It is observed that increasing the complexity and number of nodes in a well-fine-tuned model like byt5 will decrease the accuracy of link prediction in the same class of graphs, but it will increase the generalizability (refer to section 3.4). For comparison, we also conducted link prediction using the base byt5 model without fine-tuning [74] and a fine-tuned BART model [81] (refer to Appendix for additional information). Both models demonstrated suboptimal performance, failing to follow a predictable trend.

Table 8: Edge Prediction accuracy using LLMs on the test set

Case	Phase	**Link Prediction Accuracy (Base Byt5)**	**Link Prediction Accuracy (Fine-tuned BART)**	**Link Prediction Accuracy (Fine-tuned Byt5)**
Simple Graphs (Dataset 1)	Solid Phase	~0 % The base-byt5 prediction, given a prompt requesting the link, yields a highly unpredictable outcome that cannot be reverted to a graph representation.	5.42 % - mostly disconnected	98.51 %
	Pore Phase		5.49 % - mostly disconnected	98.72 %
Complex Graphs (Dataset 2)	Solid Phase		6.46 % - mostly disconnected	73.75 %
	Pore Phase		6.43 % - mostly disconnected	73.88 %
Complex Graphs (Dataset 3)	Solid Phase		4.46 % - mostly disconnected	69.53 %
	Pore Phase		5.62 % - mostly disconnected	69.41 %

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

The base byt5 model without fine-tuning produced highly unpredictable outcomes, which in most cases cannot be reverted to a meaningful graph representation. Similarly, the fine-tuned BART model, given the same amount of training data, exhibited very low accuracy and generated text that could not be converted into a desired graph representation, often resulting in disconnected graphs with low accuracies. These results highlight the importance of selecting the appropriate LLM. Henceforth, due to the reasonable prediction ability of the fine-tuned byt5 model, only this model will be used in the following steps of metamaterial unit generation. Consequently, any reference to a LLM in the remainder of the paper will pertain exclusively to our fine-tuned byt5 model. Figure 6 shows the prediction accuracy of the fine-tuned byt5 model for all samples from the test dataset for dual phases. Here, each point represents the accuracy of edge predictions for each graph sample, which encompasses multiple nodes and edges.

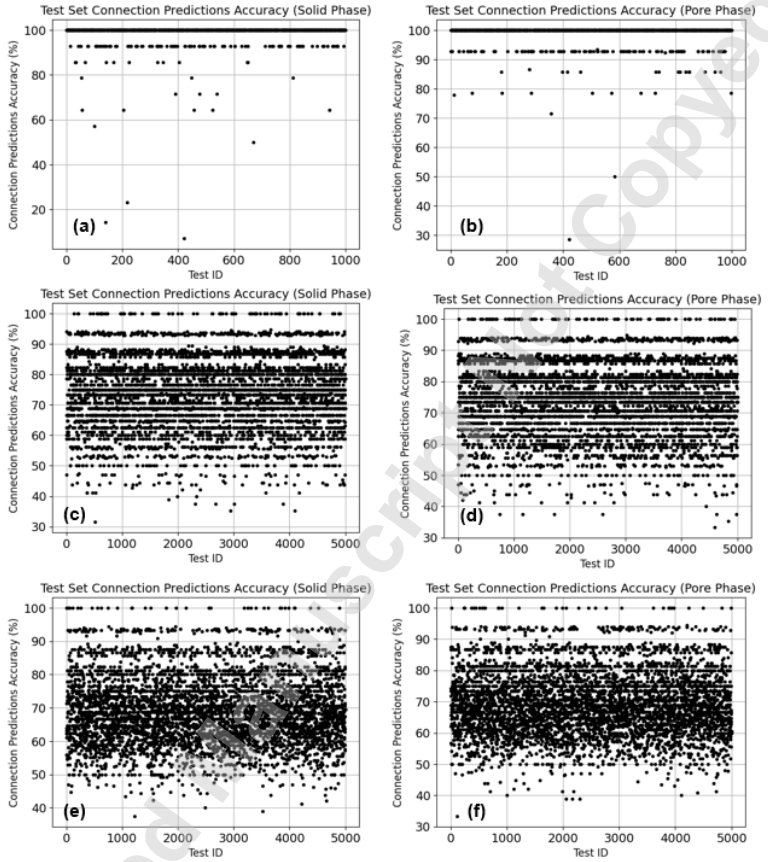


Figure 6: The fine-tuned byt5 model's prediction accuracy of test samples. (a) Solid phase graphs in Dataset 1. (b) Pore phase graphs in Dataset 1. (c) Solid phase graphs in Dataset 2. (d) Pore phase graphs in Dataset 2. (e) Solid phase graphs in Dataset 3. (f) Pore phase graphs in Dataset 3.

As depicted in Figure 6, some of the reconstructions exhibit low accuracy, which can be attributable to the inherent non-determinism of the LLM model [82]. This non-determinism occasionally results in suboptimal outcomes. This issue can be mitigated by adjusting the temperature, a hyperparameter in charge of the randomness of prediction, or implementing the synthesizer [83, 84] to validate predictions against problem requirements and detect discrepancies. This scenario is illustrated in Figure 7, where the LLM model generates three different predictions for a given set of nodes, each yielding varying accuracies.

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

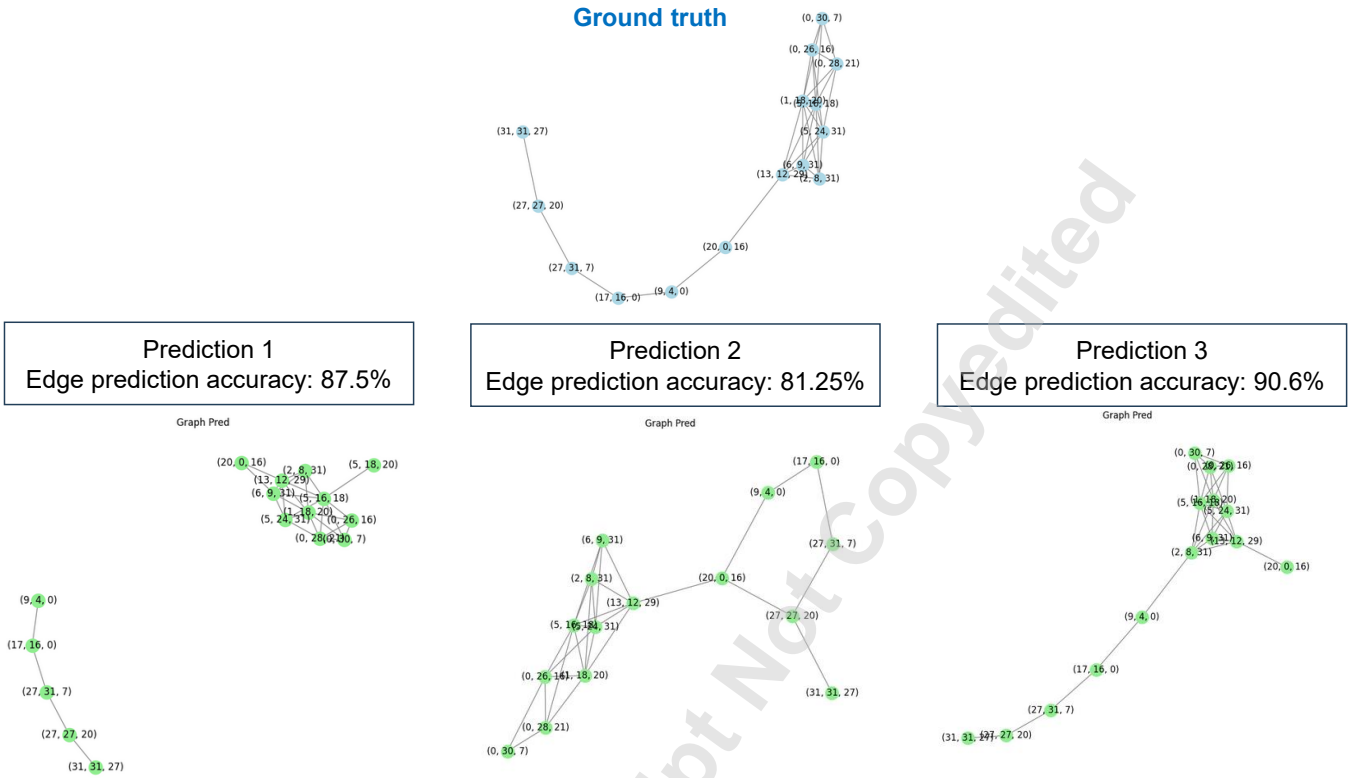


Figure 7: Illustration of LLM model's non-determinism: three realizations in predicting node connection with same set of nodes

Certain predictions, like Prediction 1 in Figure 7, might not fulfill the connectivity requirement. Therefore, a synthesizer can evaluate the problem's requirements and selectively accept predictions that meet these requirements. Nevertheless, even with a synthesizer, the model's prediction can have different accuracy, as seen in Predictions 2 and 3.

3.3 Generation of metamaterial designs in voxel format

By integrating the VGAE as node generator, the LLM as edge predictor, and voxel labeling by the approximate K-nearest neighbor-based clustering (Step 4 in Figure 2), we showcase the capability of generating voxel images of porous metamaterial samples in the training and testing dataset. The accuracy of reconstructing voxel images of porous structures is validated by comparing the reconstructions with the original image in the test set. The voxel-to-voxel reconstruction accuracy is measured by the coefficient of determination (R^2) is illustrated in Table 9.

Table 9: Voxel-to-voxel reconstruction accuracy measured by R^2 score

Coefficient of determination (R^2)		
Dataset 1	Dataset 2	Dataset 3
0.98	0.57	0.51

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

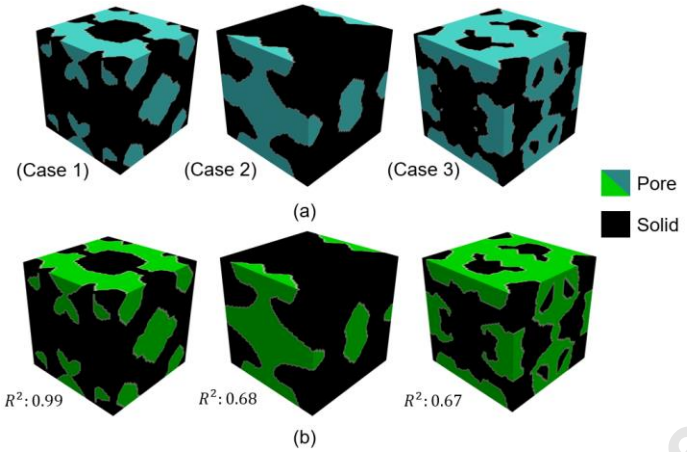


Figure 8: Reconstruction results of the proposed generative model for generating voxel images of porous metamaterials: (a) original samples in the training dataset and (b) reconstructions.

3.4 Generation of unseen porous metamaterial designs

We demonstrate that the fine-tuned LLM can connect nodes based on the logic learned from the training dataset, even when an arbitrary number of nodes is provided as input (zero-shot learning capability). To evaluate the effectiveness of zero-shot learning, the model trained on Datasets 3 has been employed. Figure 9 shows a "ground truth" graph generated according to the logic outlined in Section 2.1 but with number of nodes as 75, which is not available in the training samples (Dataset 3), alongside a graph generated by the fine-tuned LLM (Dataset 3). The graph reconstructed by LLM has 63.15% accuracy in predicting the ground truth. This achieved accuracy can be attributed to the model's enhanced generalizability [71, 72], enabling more accurate predictions of unseen data.

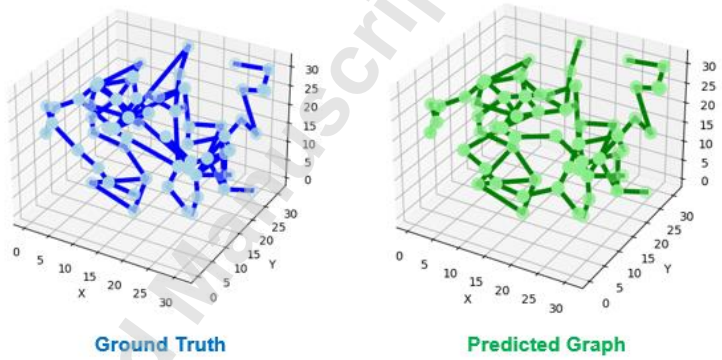


Figure 9: Demonstration of the zero-shot learning capability of a fine-tuned LLM for edge reconstruction for a sample with 75 nodes

4. DISCUSSION

The graph-based representation of porous metamaterials facilitates the utilization of both GNNs and LLMs in design generation. As demonstrated by the results of the computational experiments, GNNs, such as VGAEs, excel in reconstructing node features but may struggle to capture the underlying logic governing connections between nodes. On the other hand, Transformer-based LLMs excel in comprehending long-range dependencies [85] and diverse relationships, making them well-suited for tasks such as edge prediction in graph-based problems. Additionally, because of their multi-head attention mechanism [43], LLMs have the capability of parallel processing in linking nodes, as opposed to sequential processes like Long short-term memory (LSTM) and recurrent neural network (RNN), making LMM models faster and more efficient in capturing longer-range dependencies [86, 87]. Despite their effectiveness in detecting

relationships and ability to do zero-shot learning [50-52], the application of LLMs in structural design, including metamaterials, remains limited mainly due to their "black box" [77] and non-deterministic [82] nature. Nevertheless, the integration of LLMs and GNNs presented a promising method in metamaterial structural design [42, 44].

5. CONCLUSION

The purpose of this work is to establish an approach for generating porous metamaterial units based on certain rules, which are unknown and must be learned from an observational dataset. Based on the graph representation of porous metamaterial designs, two new approaches are proposed for the representation, reconstruction, and generation of porous metamaterials. The first approach utilizes a DVGAE for predicting both nodes and edges of the graphs representing the solid and pore phases in the porous structure. The second approach employs a VGAE as the node generator and a fine-tuned LLM as the edge generator. In the comparative study, we observe that LLM demonstrates significant strength in reconstructing the edges in graphs. The selection of an appropriate LLM is crucial for achieving high prediction accuracy. Additionally, we demonstrate the zero-shot learning capability of our proposed model by generating structural patterns that were not included in the observational dataset.

In future works, we intend to enhance the model by utilizing strategies like Low-rank adaptation [88] and parameter-efficient fine-tuning [89, 90]. These methods aim to boost the model's accuracy and enable the use of a more complex and efficient model than current LLMs. Additionally, the current model is not invariant to ordering. We plan to investigate further to achieve ordering invariance, which may enable us to create a new synthesizer, resulting in predictions with higher accuracy. We will also establish a porous metamaterial design framework based on the deep generative models proposed in this work.

ACKNOWLEDGEMENT

We gratefully acknowledge the financial support from the National Science Foundation (CMMI-2142290). We express our gratitude to Dr. Yewen Pu for the tutorial on LLMs at the Second Frontiers of Design Representation Summer School hosted by the University of Maryland.

Journal of Computing and Information Science in Engineering. Received April 05, 2024;
Accepted manuscript posted July 16, 2024. doi:10.1115/1.4066095

Appendix

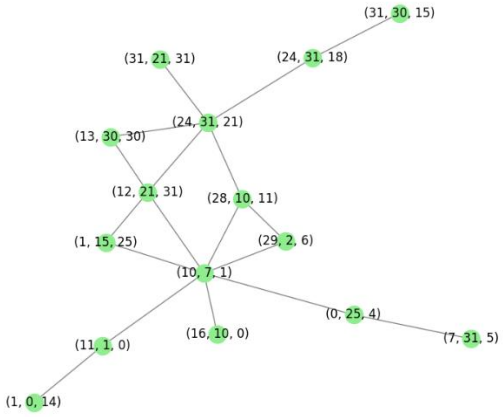
Here we introduce a second LLM for comparison with the fine-tuned Byt5. As suggested in section 2.4, the encoder-decoder framework is ideal for seq2seq tasks. Therefore, BART, which is a denoising autoencoder designed to pretrain on sequence-to-sequence tasks and incorporates the encoder-decoder framework, has been chosen. BART utilizes a bidirectional decoder and an autoregressive decoder [78]. The process of fine-tuning this model is similar to the Byt5 model in Figure 4. The base model and tokenizer are available in the hugging face's Bart-base repository [81]. Table 10 demonstrates a graph ground truth, predicted output, and extracted graph via both models.

Table 10: Comparison between the encoder-decoder structured LLMs

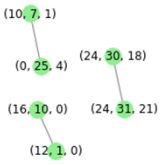
Ground Truth

LLM Input Nodes	
['(24, 31, 21)', '(24, 31, 18)', '(31, 21, 31)', '(12, 21, 31)', '(13, 30, 30)', '(28, 10, 11)', '(31, 30, 15)', '(11, 1, 0)', '(10, 7, 1)', '(1, 0, 14)', '(16, 10, 0)', '(0, 25, 4)', '(29, 2, 6)', '(1, 15, 25)', '(7, 31, 5)']	

Byt5

Byt5 output	
['24;31;21&24;31;18;;24;31;21&31;21;31;;24;31;21&12;21;31;;24;31; 21&13;30;30;;24;31;21&28;10;11;;24;31;18&31;30;15;;12;21;31&10; 7;1;;12;21;31&13;30;30;;12;21;31&1;15;25;;28;10;11&10;7;1;;28;10; 11&29;2;6;;11;1;0&10;7;1;;11;1;0&1;0;14;;10;7;1&16;10;0;;10;7;1& 0;25;4;;10;7;1&29;2;6;;10;7;1&1;15;25;;0;25;4&7;31;5;;']	
Predicted Graph (Accuracy = 77.77%)	
[((24, 31, 21), (24, 31, 18)), ((24, 31, 21), (31, 21, 31)), ((24, 31, 21), (12, 21, 31)), ((24, 31, 21), (13, 30, 30)), ((24, 31, 21), (28, 10, 11)), ((24, 31, 18), (31, 30, 15)), ((12, 21, 31), (10, 7, 1)), ((12, 21, 31), (13, 30, 30)), ((12, 21, 31), (1, 15, 25)), ((28, 10, 11), (10, 7, 1)), ((28, 10, 11), (29, 2, 6)), ((10, 7, 1), (11, 1, 0)), ((10, 7, 1), (16, 10, 0)), ((10, 7, 1), (0, 25, 4)), ((10, 7, 1), (29, 2, 6)), ((10, 7, 1), (1, 15, 25)), ((11, 1, 0), (1, 0, 14)), ((0, 25, 4), (7, 31, 5))]	

BART

BART output	
['24;31;21&24;30;18;;24;29;21;21;;26;31;;31;18;31&12;21 wing31;;12;1;0&16;10;0;;10;7;1&0;25;4;;10&29;7&28;27;15;;0;15;2 5&1;15 considerably25;;']	
Predicted Graph (Accuracy = 0%)	
[((24, 31, 21), (24,30,18)), ((12, 1, 0), (16, 10, 0)), ((10,7,1), (0,25,4))]	

REFERENCES

1. Zheng, X., H. Lee, T.H. Weisgraber, M. Shusteff, J. DeOtte, E.B. Duoss, J.D. Kuntz, M.M. Biener, Q. Ge, and J.A. Jackson, *Ultralight, ultrastiff mechanical metamaterials.* Science, 2014. **344**(6190): p. 1373-1377.
2. Chen, H. and C.T. Chan, *Acoustic cloaking in three dimensions using acoustic metamaterials.* Applied physics letters, 2007. **91**(18): p. 183518.
3. Garland, A.P., K.M. Adstedt, Z.J. Casias, B.C. White, W.M. Mook, B. Kaehr, B.H. Jared, B.T. Lester, N.S. Leathe, and E. Schwaller, *Coulombic friction in metamaterials to dissipate mechanical energy.* Extreme Mechanics Letters, 2020. **40**: p. 100847.
4. Claeys, C., N.G.R. de Melo Filho, L. Van Belle, E. Deckers, and W. Desmet, *Design and validation of metamaterials for multiple structural stop bands in waveguides.* Extreme Mechanics Letters, 2017. **12**: p. 7-22.
5. Qian, J., Y. Cheng, A. Zhang, Q. Zhou, and J. Zhang, *Optimization design of metamaterial vibration isolator with honeycomb structure based on multi-fidelity surrogate model.* Structural and Multidisciplinary Optimization, 2021. **64**: p. 423-439.
6. Wang, Z., W. Xian, M.R. Baccouche, H. Lanzerath, Y. Li, and H. Xu, *Design of Phononic Bandgap Metamaterials based on Gaussian Mixture Beta Variational Autoencoder and Iterative Model Updating.* Journal of Mechanical Design, 2022: p. 1-35.
7. Wang, Z., W. Xian, M.R. Baccouche, H. Lanzerath, Y. Li, and H. Xu. *A Gaussian Mixture Variational Autoencoder-Based Approach for Designing Phononic Bandgap Metamaterials*. in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. 2021. American Society of Mechanical Engineers.
8. Wang, Z., R. Zhuang, W. Xian, J. Tian, Y. Li, S. Chen, and H. Xu. *Phononic Metamaterial Design via Transfer Learning-Based Topology Optimization Framework*. in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. 2022. American Society of Mechanical Engineers.
9. Gurbuz, C., F. Kronowetter, C. Dietz, M. Eser, J. Schmid, and S. Marburg, *Generative adversarial networks for the design of acoustic metamaterials.* J Acoust Soc Am, 2021. **149**(2): p. 1162.
10. Alberdi, R., R. Dingreville, J. Robbins, T. Walsh, B.C. White, B. Jared, and B.L. Boyce, *Multi-morphology lattices lead to improved plastic energy absorption.* Materials & Design, 2020. **194**: p. 108883.
11. Xu, H. and Z. Liu, *Control variate multifidelity estimators for the variance and sensitivity analysis of mesostructure–structure systems.* ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering, 2019. **5**(2): p. 020907.
12. Liu, Z., H. Xu, and P. Zhu, *An adaptive multi-fidelity approach for design optimization of mesostructure-structure systems.* Structural and Multidisciplinary Optimization, 2020. **62**: p. 375-386.
13. Zhang, Q., K. Zhang, and G. Hu, *Tunable fluid-solid metamaterials for manipulation of elastic wave propagation in broad frequency range.* Applied Physics Letters, 2018. **112**(22).
14. He, Z.-H., Y.-Z. Wang, and Y.-S. Wang, *Active feedback control of sound radiation in elastic wave metamaterials immersed in water with fluid–solid coupling.* Acta Mechanica Sinica, 2021. **37**: p. 803-825.
15. Song, Y. and Y. Shen, *Highly morphing and reconfigurable fluid–solid interactive metamaterials for tunable ultrasonic guided wave control.* Applied Physics Letters, 2022. **121**(26).
16. Gao, D., J. Chen, Z. Dong, and H. Lin, *Connectivity-guaranteed porous synthesis in free form model by persistent homology.* Computers & Graphics, 2022. **106**: p. 33-44.
17. Swartz, K.E., D.A. Tortorelli, D.A. White, and K.A. James, *Manufacturing and stiffness constraints for topology optimized periodic structures.* Structural and Multidisciplinary Optimization, 2022. **65**(4): p. 129.
18. Holdstein, Y., A. Fischer, L. Podshivalov, and P.Z. Bar-Yoseph. *Volumetric texture synthesis of bone micro-structure as a base for scaffold design*. in *2009 IEEE international conference on shape modeling and applications*. 2009. IEEE.
19. Men, H., K.Y. Lee, R.M. Freund, J. Peraire, and S.G. Johnson, *Robust topology optimization of three-dimensional photonic-crystal band-gap structures.* Optics express, 2014. **22**(19): p. 22632-22648.
20. Kench, S. and S.J. Cooper, *Generating 3D structures from a 2D slice with GAN-based dimensionality expansion.* arXiv preprint arXiv:2102.07708, 2021.
21. Zheng, X., X. Guo, Y. Yang, Z. Fu, K. Du, C. Wang, and Y. Yi, *Structure-dependent analysis of nanoporous metals: clues from mechanical, conduction, and flow properties.* The Journal of Physical Chemistry C, 2018. **122**(29): p. 16803-16809.

22. Xu, H., D.A. Dikin, C. Burkhart, and W. Chen, *Descriptor-based methodology for statistical characterization and 3D reconstruction of microstructural materials.* Computational Materials Science, 2014. **85**: p. 206-216.
23. Meyer, P.P., C. Bonatti, T. Tancogne-Dejean, and D. Mohr, *Graph-based metamaterials: Deep learning of structure-property relations.* Materials & Design, 2022. **223**: p. 111175.
24. Makatura, L., B. Wang, Y.-L. Chen, B. Deng, C. Wojtan, B. Bickel, and W. Matusik, *Procedural metamaterials: A unified procedural graph for metamaterial design.* ACM Transactions on Graphics, 2023. **42**(5): p. 1-19.
25. Yamaguchi, K., H. Yasuda, K. Tsujikawa, T. Kunimine, and J. Yang, *Graph-theoretic estimation of reconfigurability in origami-based metamaterials.* Materials & Design, 2022. **213**: p. 110343.
26. Du, P., A. Zebrowski, J. Zola, B. Ganapathysubramanian, and O. Wodo, *Microstructure design using graphs.* npj Computational Materials, 2018. **4**(1): p. 50.
27. Guo, K. and M.J. Buehler, *A semi-supervised approach to architected materials design using graph neural networks.* Extreme Mechanics Letters, 2020. **41**: p. 101029.
28. Reiser, P., M. Neubert, A. Eberhard, L. Torresi, C. Zhou, C. Shao, H. Metni, C. van Hoesel, H. Schopmans, and T. Sommer, *Graph neural networks for materials science and chemistry.* Communications Materials, 2022. **3**(1): p. 93.
29. Nourian, N., M. El-Badry, and M. Jamshidi, *Design Optimization of Truss Structures Using a Graph Neural Network-Based Surrogate Model.* Algorithms, 2023. **16**(8): p. 380.
30. Prachaseree, P. and E. Lejeune, *Learning mechanically driven emergent behavior with message passing neural networks.* Computers & Structures, 2022. **270**: p. 106825.
31. Indurkar, P.P., S. Karlapati, A.J.D. Shaikeea, and V.S. Deshpande, *Predicting deformation mechanisms in architected metamaterials using GNN.* arXiv preprint arXiv:2202.09427, 2022.
32. Maurizi, M., C. Gao, and F. Berto, *Predicting stress, strain and deformation fields in materials and structures with graph neural networks.* Scientific Reports, 2022. **12**(1): p. 21834.
33. Ross, E. and D. Hambleton, *Using graph neural networks to approximate mechanical response on 3d lattice structures.* Proceedings of AAG2020-Advances in Architectural Geometry, 2021. **24**: p. 466-485.
34. Wang, Z., Bray, Z., Naghavi Khanghah, K., Xu, H., *A Generative Graph Neural Network-based Framework for Designing Connectivity-Guaranteed Porous Metamaterial Units*, in *ASME 2024 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference, (DETC2023-114601).American Society of Mechanical Engineers*. 2024, submitted.
35. Scarselli, F., M. Gori, A.C. Tsoi, M. Hagenbuchner, and G. Monfardini, *The graph neural network model.* IEEE transactions on neural networks, 2008. **20**(1): p. 61-80.
36. Wu, Z., S. Pan, F. Chen, G. Long, C. Zhang, and S.Y. Philip, *A comprehensive survey on graph neural networks.* IEEE transactions on neural networks and learning systems, 2020. **32**(1): p. 4-24.
37. Dold, D. and D.A. van Egmond, *Differentiable graph-structured models for inverse design of lattice materials.* arXiv preprint arXiv:2304.05422, 2023.
38. Zhang, M., *Graph neural networks: link prediction.* Graph Neural Networks: Foundations, Frontiers, and Applications, 2022: p. 195-223.
39. Zhang, M. and Y. Chen, *Link prediction based on graph neural networks.* Advances in neural information processing systems, 2018. **31**.
40. Kipf, T.N. and M. Welling, *Variational graph auto-encoders.* arXiv preprint arXiv:1611.07308, 2016.
41. Guo, Z., F. Wang, K. Yao, J. Liang, and Z. Wang. *Multi-scale variational graph autoencoder for link prediction*. in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 2022.
42. Jin, B., G. Liu, C. Han, M. Jiang, H. Ji, and J. Han, *Large language models on graphs: A comprehensive survey.* arXiv preprint arXiv:2312.02783, 2023.
43. Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, and I. Polosukhin, *Attention is all you need.* Advances in neural information processing systems, 2017. **30**.
44. Li, Y., Z. Li, P. Wang, J. Li, X. Sun, H. Cheng, and J.X. Yu, *A survey of graph meets large language model: Progress and future directions.* arXiv preprint arXiv:2311.12399, 2023.
45. Xie, H., D. Zheng, J. Ma, H. Zhang, V.N. Ioannidis, X. Song, Q. Ping, S. Wang, C. Yang, and Y. Xu, *Graph-Aware Language Model Pre-Training on a Large Graph Corpus Can Help Multiple Graph Applications.* arXiv preprint arXiv:2306.02592, 2023.
46. Wen, Z. and Y. Fang, *Prompt tuning on graph-augmented low-resource text classification.* arXiv preprint arXiv:2307.10230, 2023.

47. Chandra, S., P. Mishra, H. Yannakoudakis, M. Nimishakavi, M. Saeidi, and E. Shutova, *Graph-based modeling of online communities for fake news detection.* arXiv preprint arXiv:2008.06274, 2020.
48. Zhao, H., S. Liu, C. Ma, H. Xu, J. Fu, Z. Deng, L. Kong, and Q. Liu, *Gimlet: A unified graph-text model for instruction-based molecule zero-shot learning. bioRxiv*. 2023a.
49. Liu, P., Y. Ren, and Z. Ren, *GIT-Mol: A Multi-modal Large Language Model for Molecular Science with Graph.* Image, and Text, 2023.
50. Wang, T., A. Roberts, D. Hesslow, T. Le Scao, H.W. Chung, I. Beltagy, J. Launay, and C. Raffel. *What language model architecture and pretraining objective works best for zero-shot generalization?* in *International Conference on Machine Learning*. 2022. PMLR.
51. Xian, Y., B. Schiele, and Z. Akata. *Zero-shot learning-the good, the bad and the ugly*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
52. Li, J., K. Pan, Z. Ge, M. Gao, H. Zhang, W. Ji, W. Zhang, T.-S. Chua, S. Tang, and Y. Zhuang. *Fine-tuning multimodal llms to follow zero-shot demonstrative instructions*. in *The Twelfth International Conference on Learning Representations*. 2023.
53. Perozzi, B., B. Fatemi, D. Zelle, A. Tsitsulin, M. Kazemi, R. Al-Rfou, and J. Halcrow, *Let Your Graph Do the Talking: Encoding Structured Data for LLMs.* arXiv preprint arXiv:2402.05862, 2024.
54. He, X., X. Bresson, T. Laurent, A. Perold, Y. LeCun, and B. Hooi. *Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning*. in *The Twelfth International Conference on Learning Representations*. 2023.
55. Szabo, F., *The linear algebra survival guide: illustrated with Mathematica*. 2015: Academic Press.
56. Otair, D.M., *Approximate k-nearest neighbour based spatial clustering using kd tree.* arXiv preprint arXiv:1303.1951, 2013.
57. Das, K.C., A.D. Maden, I.N. Cangül, and A.S. Çevik, *On average eccentricity of graphs.* Proceedings of the National Academy of Sciences, India Section A: Physical Sciences, 2017. **87**: p. 23-30.
58. Thomas, N.K. and M. Welling, *Variational graph auto-encoders.* arXiv preprint arXiv:1611.07308, 2016. **2**(10).
59. Pan, S., R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, *Adversarially regularized graph autoencoder for graph embedding.* arXiv preprint arXiv:1802.04407, 2018.
60. Wang, C., S. Pan, R. Hu, G. Long, J. Jiang, and C. Zhang, *Attributed graph clustering: A deep attentional embedding approach.* arXiv preprint arXiv:1906.06532, 2019.
61. Sun, D., D. Li, Z. Ding, X. Zhang, and J. Tang, *Dual-decoder graph autoencoder for unsupervised graph representation learning.* Knowledge-Based Systems, 2021. **234**: p. 107564.
62. Kingma, D.P. and M. Welling, *Auto-encoding variational bayes.* arXiv preprint arXiv:1312.6114, 2013.
63. Xu, K., L. Wu, Z. Wang, Y. Feng, M. Witbrock, and V. Sheinin, *Graph2seq: Graph to sequence learning with attention-based neural networks.* arXiv preprint arXiv:1804.00823, 2018.
64. Sarkar, S. and L. Lausen, *Testing the limits of unified seq2seq LLM pretraining on diverse table data tasks.* 2023.
65. Xue, L., A. Barua, N. Constant, R. Al-Rfou, S. Narang, M. Kale, A. Roberts, and C. Raffel, *Byt5: Towards a token-free future with pre-trained byte-to-byte models.* Transactions of the Association for Computational Linguistics, 2022. **10**: p. 291-306.
66. Pu, Y., *fine-tuned llm as program writers*, in https://evantheboucy.github.io/program-synthesis-minimal/generation-with-llm/. 2022.
67. Zhang, H., H. Song, S. Li, M. Zhou, and D. Song, *A survey of controllable text generation using transformer-based pre-trained language models.* ACM Computing Surveys, 2023. **56**(3): p. 1-37.
68. Raffel, C., N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P.J. Liu, *Exploring the limits of transfer learning with a unified text-to-text transformer.* The Journal of Machine Learning Research, 2020. **21**(1): p. 5485-5551.
69. Fu, Z., W. Lam, Q. Yu, A.M.-C. So, S. Hu, Z. Liu, and N. Collier, *Decoder-Only or Encoder-Decoder? Interpreting Language Model as a Regularized Encoder-Decoder.* arXiv preprint arXiv:2304.04052, 2023.
70. Pourpanah, F., M. Abdar, Y. Luo, X. Zhou, R. Wang, C.P. Lim, X.-Z. Wang, and Q.J. Wu, *A review of generalized zero-shot learning methods.* IEEE transactions on pattern analysis and machine intelligence, 2022.
71. Zhang, D., J. Wang, and F. Charton, *Instruction Diversity Drives Generalization To Unseen Tasks.* arXiv preprint arXiv:2402.10891, 2024.
72. Tang, J., Y. Yang, W. Wei, L. Shi, L. Su, S. Cheng, D. Yin, and C. Huang, *Graphgpt: Graph instruction tuning for large language models.* arXiv preprint arXiv:2310.13023, 2023.

73. Shen, Z., M. Zhang, H. Zhao, S. Yi, and H. Li. *Efficient attention: Attention with linear complexities*. in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2021.
74. Huggingface. *google/byt5-base*. Available from: https://huggingface.co/google/byt5-base.
75. Huggingface. *Seq2Seq-Trainer*. Available from: https://github.com/huggingface/transformers/tree/main/examples/legacy/seq2seq.
76. Hagberg, A., P.J. Swart, and D.A. Schult, *Exploring network structure, dynamics, and function using NetworkX*. 2008, Los Alamos National Laboratory (LANL), Los Alamos, NM (United States).
77. Bhattacharjee, A., R. Moraffah, J. Garland, and H. Liu, *Towards LLM-guided Causal Explainability for Black-box Text Classifiers.* 2024.
78. Lewis, M., Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, *Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.* arXiv preprint arXiv:1910.13461, 2019.
79. Renaud, O. and M.-P. Victoria-Feser, *A robust coefficient of determination for regression.* Journal of Statistical Planning and Inference, 2010. **140**(7): p. 1852-1862.
80. Helland, I.S., *On the interpretation and use of R2 in regression analysis.* Biometrics, 1987: p. 61-69.
81. Huggingface. *Facebook/Bart-base*. Available from: https://huggingface.co/facebook/bart-base.
82. Ouyang, S., J.M. Zhang, M. Harman, and M. Wang, *LLM is Like a Box of Chocolates: the Non-determinism of ChatGPT in Code Generation.* arXiv preprint arXiv:2308.02828, 2023.
83. Pu, Y., K. Ellis, M. Kryven, J. Tenenbaum, and A. Solar-Lezama, *Program synthesis with pragmatic communication.* Advances in Neural Information Processing Systems, 2020. **33**: p. 13249-13259.
84. Pu, Y., Z. Miranda, A. Solar-Lezama, and L.P. Kaelbling, *Learning to select examples for program synthesis.* 2018.
85. Zhang, C., T. Lu, M.M. Islam, Z. Wang, S. Yu, M. Bansal, and G. Bertasius, *A simple llm framework for long-range video question-answering.* arXiv preprint arXiv:2312.17235, 2023.
86. Raiaan, M.A.K., M.S.H. Mukta, K. Fatema, N.M. Fahad, S. Sakib, M.M.J. Mim, J. Ahmad, M.E. Ali, and S. Azam, *A review on large Language Models: Architectures, applications, taxonomies, open issues and challenges.* IEEE Access, 2024.
87. Liu, X., H.-F. Yu, I. Dhillon, and C.-J. Hsieh. *Learning to encode position for transformer with continuous dynamical model*. in *International conference on machine learning*. 2020. PMLR.
88. Hu, E.J., Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, *Lora: Low-rank adaptation of large language models.* arXiv preprint arXiv:2106.09685, 2021.
89. Liu, H., D. Tam, M. Muqeeth, J. Mohta, T. Huang, M. Bansal, and C.A. Raffel, *Few-shot parameter-efficient fine-tuning is better and cheaper than in-context learning.* Advances in Neural Information Processing Systems, 2022. **35**: p. 1950-1965.
90. Mo, Y., J. Yoo, and S. Kang, *Parameter-Efficient Fine-Tuning Method for Task-Oriented Dialogue Systems.* Mathematics, 2023. **11**(14): p. 3048.