

IDETC2024-143234

GENERATING POROUS METAMATERIAL DESIGNS USING 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 application needs [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 effectiveness 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 and connection information $\mathbf{G} = (\mathbf{V}, \mathbf{E})$, 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, the 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 LLM 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 the 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 2.4, 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 2.1 and 2.2, 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 2.3, 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 2.4 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.

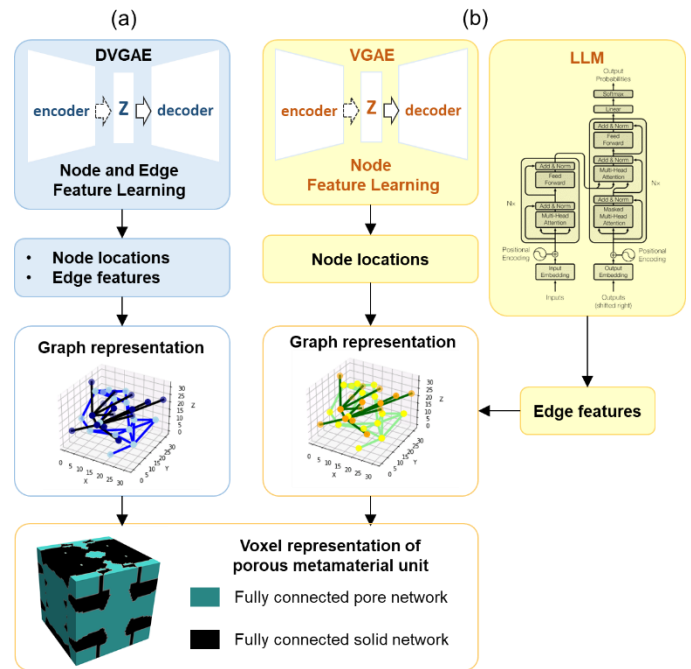


Figure 1: Proposed approaches for generating complex porous metamaterial designs. (a) Dual Decoder Variational Graph Autoencoder (DVGAE) for generating both nodes and edges, and (b) A hybrid approach that integrates VGAE and LLM (LLM model architecture presented by Vaswani et al. [43]).

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

Graph autoencoder (GAE) and its variations (e.g., VGAE [55], adversarial regularized graph autoencoder [56], deep attention embedding graph autoencoder [57], etc.) learn latent representations of graphs using an autoencoder framework has been widely used to generate new graphs. DVGAE [58] 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:

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

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

where we define an undirected, unweighted graph $G = (v, \varepsilon)$ with $N = |v|$ nodes. $\mathbf{X}$ is the node features matrix of graph G , $\mathbf{A}$ is the adjacency matrix of 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 $N \times D$. $\mathcal{N}(\cdot)$ stands for the normal distribution. 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 ($\boldsymbol{\mu} = \text{GCN}_{\boldsymbol{\mu}}(\mathbf{X}, \mathbf{A})$) and another for the log variance ($\log \boldsymbol{\sigma} = \text{GCN}_{\boldsymbol{\sigma}}(\mathbf{X}, \mathbf{A})$). 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 $\tilde{\mathbf{A}}$ is reconstructed as:

$$\tilde{\mathbf{A}} = \sigma(\mathbf{z}\mathbf{z}^T) \quad (3)$$

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 graph convolutional layer followed by a node-wise softmax operation to reconstruct node features $\tilde{\mathbf{X}}$. The graph convolutional decoder is defined as:

$$\tilde{\mathbf{X}} = f(\mathbf{z}, \mathbf{A}) = \underline{\mathbf{A}} \text{ReLU}(\underline{\mathbf{A}}\mathbf{z}\mathbf{W}^{(0)})\mathbf{W}^{(1)} \quad (4)$$

where $\mathbf{A}$ is the adjacency matrix. $\mathbf{z}$ represents the latent representation obtained from encoder. $\text{ReLU}(\cdot) = (0; \cdot)$ 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 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:

$$\mathcal{L}_{adj} = \mathbb{E}_{q(\mathbf{z}|\mathbf{X}, \mathbf{A})} [\log p(\mathbf{A}|\mathbf{z})] \quad (5)$$

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:

$$p(\mathbf{A}|\mathbf{z}) = \prod_{i=1}^N \prod_{j=1}^N p(A_{ij}|\mathbf{z}_i, \mathbf{z}_j) \quad (6)$$

$$p(A_{ij} = 1|\mathbf{z}_i, \mathbf{z}_j) = \sigma(\mathbf{z}_i^T \mathbf{z}_j) \quad (7)$$

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

$$\mathcal{L}_X = \frac{1}{2} \|\mathbf{X} - \tilde{\mathbf{X}}\|^2 \quad (8)$$

The VGAE model is trained to optimize the variational lower bound L , which is the sum of reconstruction loss for adjacency matrix (Eq.5), reconstruction loss for node features (Eq.8) and the Kullback-Leibler divergence between $q(\cdot)$ and $p(\cdot)$:

$$\begin{aligned} \mathcal{L}_{VGAE} &= \lambda_1 \mathcal{L}_{adj} + \lambda_2 \mathcal{L}_X + \lambda_3 \mathcal{L}_{KL} = \\ &\lambda_1 \mathbb{E}_{q(\mathbf{z}|\mathbf{X}, \mathbf{A})} [\log p(\mathbf{A}|\mathbf{z})] + \lambda_2 \frac{1}{2} \|\mathbf{X} - \tilde{\mathbf{X}}\|^2 - \\ &\lambda_3 \text{KL}[q(\mathbf{z}|\mathbf{X}, \mathbf{A})||p(\mathbf{z})] \end{aligned} \quad (9)$$

We use Gaussian prior $p(\mathbf{z}) = \prod_i p(\mathbf{z}_i) = \prod_i \mathcal{N}(\mathbf{z}_i|0, \mathbf{I})$. To optimize the parameters of the Gaussian distribution, we perform mini-batch gradient descent and leverage the reparameterization trick [59]. λ_1 , λ_2 and λ_3 are coefficients to balance different loss terms to achieve better accuracy. In this work, we use $\lambda_1 = \lambda_2 = \lambda_3 = 1$ for simplicity. Although we experimented with other combinations of λ_1 , λ_2 and λ_3 for hyperparameter tuning, we did not observe significant improvements by varying these values.

2.2 DGVAE 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. The porous metamaterial dataset (see Section 2.4) is divided into two sets, 222,937 for training and 3,000 for testing. Adam is used as the optimizer for parameter optimization.

2.3 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.

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 2.4, 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 reasoning 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 [60, 61] is employed to get nodes' positional information as input and predict the nodes' connections as the output.

In this paper, the ByT5-base from the hugging face library has been utilized for edge prediction [62]. 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 [63]. ByT5, similar to other variants of the Text-to-Text Transfer Transformer (T5) model, has an Encoder-Decoder architecture [64]. The encoder-decoder framework is well-suited for seq2seq tasks due to its capability to maintain effective attention on both source and target sequences [65]. 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, 66].

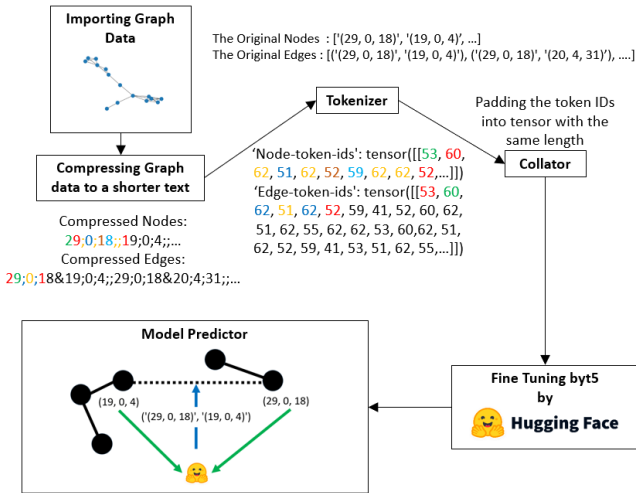


Figure 2: The approach for fine-tuning the byt5-base model for the task of predicting node connections in graphs that represent the solid and pore phases in porous metamaterials. (The Hugging Face logo is provided by the Hugging Face: <https://huggingface.co/brand>)

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. [67], 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. In this study, 222,937 porous metamaterial samples in graph representation are used to fine-tune the byt5 model. The remaining 3,000 data are used for validation. In the proposed approach, as depicted in Figure 2, the following steps are undertaken.

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[62]. This compression is necessary to increase performance when leveraging attention mechanisms, which typically have a complexity of $O(n^2)$ [43, 62, 68].

The compressed data will be turned into tokens via the byt5 special tokenizer in which each byte of compressed data will be turned into a token. These tokens after getting converted into tensors of the same length via collator, undergo fine-tuning in the byt5-base model using the seq2seq trainer from the hugging face library [62]. During this 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 2.4).

The objective of the fine-tuned model is to predict node connectivity by feeding nodal information to the large language model, 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. Figure 3 presents a comparison between the graph predicted by a 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.

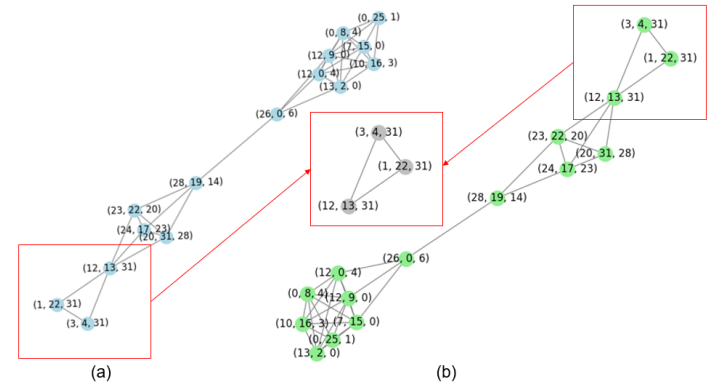


Figure 3: Demonstrating Graph Prediction: (a) A ground truth graph from the test set and (b) the predicted graph by LLM (graph is inverted).

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, while 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 2.4, 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.

2.4. 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 4, this approach consists of the following four steps.

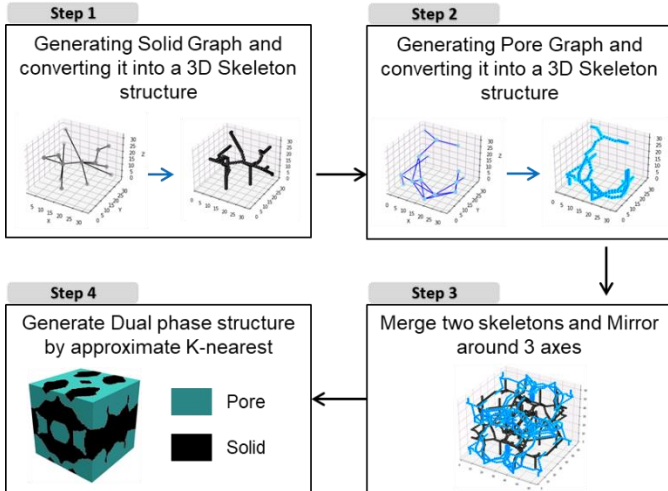


Figure 4: The proposed approach for generating complex porous metamaterial unit samples with complete connectivity in both solid and pore phases-demonstrating using one of the most complex samples.

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 case of isolated node cluster, long range connections will be established to achieve a single connected graph. The nodes and edges of the graph are then mapped to 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 [69] is conducted to bypass the blocking voxel.

Thirdly, 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) [70], 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 is not required to perform any additional post-processing to remove “enclosed voids” or “disconnected materials”. By randomizing the locations of the input nodes, a highly diverse metamaterial unit database with 225,937 samples is generated (Figure 5).

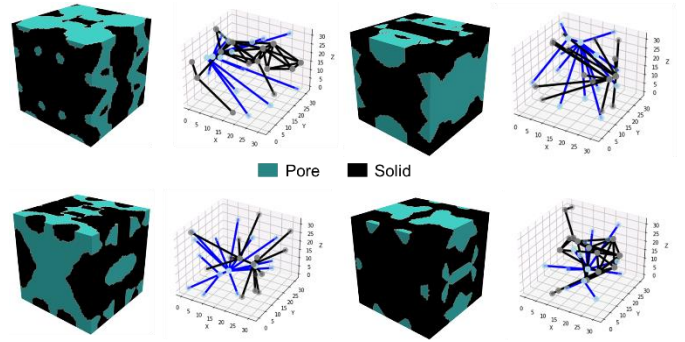


Figure 5: Diversity of the training samples: several examples of metamaterial units, the voxel image and the graph representation of 1/8 of the cube, in the created database.

3. RESULTS OF COMPARATIVE STUDIES

3.1 Reconstruction Accuracy of the DGVAE Model

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

$$R^2 = 1 - \frac{\sum (Y_i - \tilde{Y}_i)^2}{\sum (Y_i - \bar{Y})^2} \quad (10)$$

where Y_i represents the true response of the i^{th} sample, $\tilde{Y}_i$ represents the predicted response of the i^{th} sample, and n_{sample} represents the total number of sample points. $\bar{Y}$ is the averaged value of $Y_{i(true)}$ and $\bar{Y}_i = \frac{1}{n_{sample}} \sum Y_i$. 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 model predicted correctly which matches the actual presence of an edge in the graph from test dataset. The reconstruction accuracies of both solid phase and pore phase graphs are shown in Table 1.

The results reveal that DVGAE successfully reconstructs $\tilde{\mathbf{X}}$, but fails to reconstruct $\tilde{\mathbf{A}}$ accurately. The failure of reconstruction of $\tilde{\mathbf{A}}$ 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 2.4, long-range connections are also established between isolated node clusters. Thus, DVGAE struggles to capture these intricate rules, resulting in low link prediction accuracy.

Table 1: Reconstruction accuracies of both solid phase and pore phase. R^2 quantifies the reconstruction accuracy of $\tilde{\mathbf{X}}$ and $\tilde{\mathbf{A}}$. Link prediction accuracy is measured as the percentage of correctly predicted edge pairs matching the test dataset.

Phase		$\tilde{\mathbf{X}}$	$\tilde{\mathbf{A}}$	Link Prediction Accuracy
Solid	Training	0.999	-1.520	10.23%
	Test	0.999	-6.801	5.05%
Pore	Training	0.999	-1.424	10.86%
	Test	0.999	-6.491	5.12%

3.2 Reconstruction accuracy of the LLM Model

Since node-based techniques like DGVAE have poor accuracy when it comes to edge prediction, we tested edge creation using the proposed LLM model. The edge prediction accuracies of both solid and pore phase graphs are shown in Table 2.

In this method, the average accuracy of 3,000 test samples for solid phase is 95.19%, and for the pore phase is 94.41 %. Figure 6 shows the prediction accuracy for all 3,000 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.

Table 2: Link Prediction accuracy using LLM on the test set. Link prediction accuracy is measured as the percentage of correctly predicted edge pairs matching the test dataset.

Phase	Link Prediction Accuracy
Solid Phase	95.19 %
Pore Phase	94.41 %

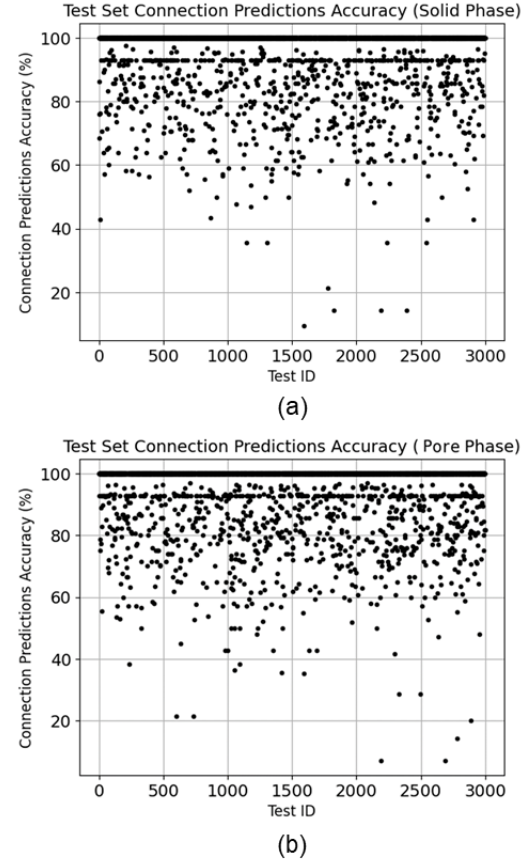


Figure 6: The prediction accuracy for 3,000 test samples (a) Solid phase (b) Pore phase.

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 [71]. 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 [72, 73] 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. Certain predictions, like Prediction 1 in Figure 7 (next page), 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.

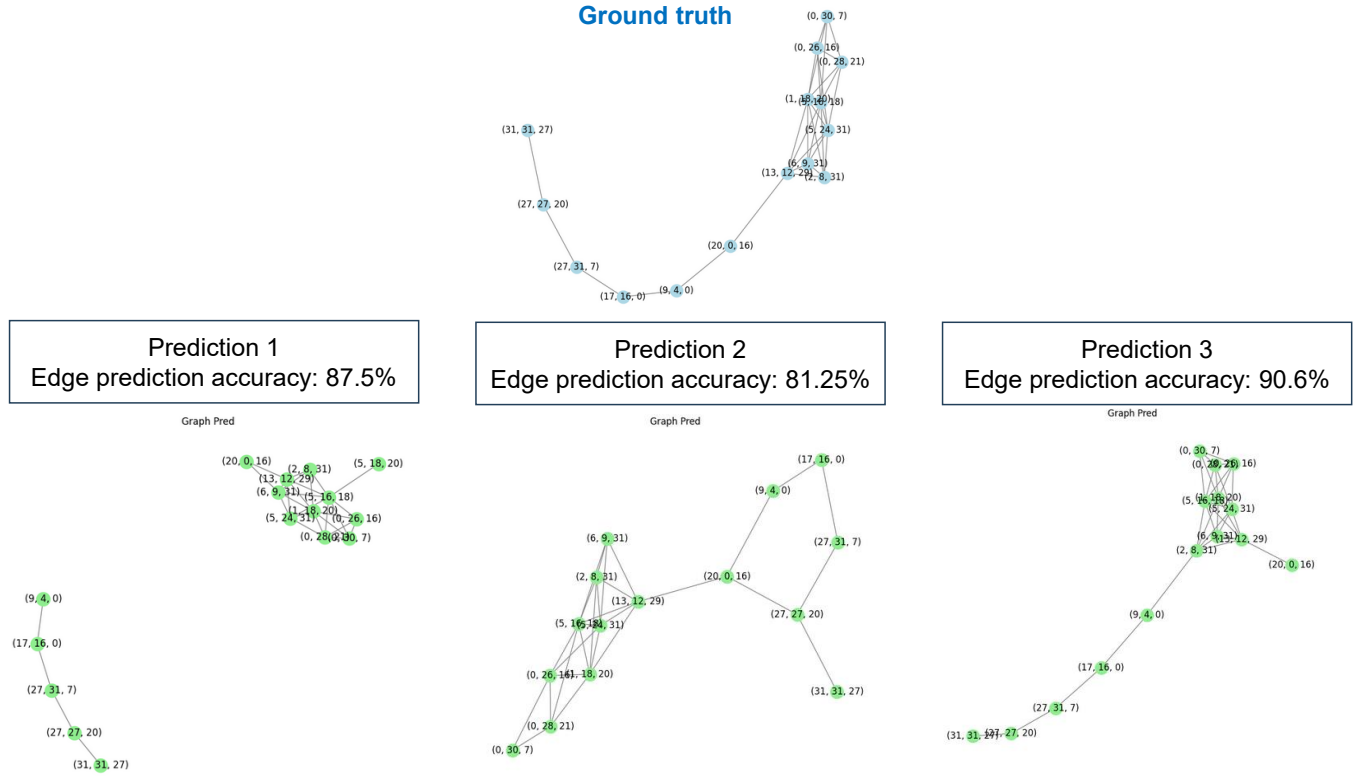


Figure 7: Illustration of LLM model's non-determinism: three realizations in predicting node connection with same set of nodes

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 4), we showcase the capability of generating voxel images of porous metamaterial samples in the training and testing dataset.

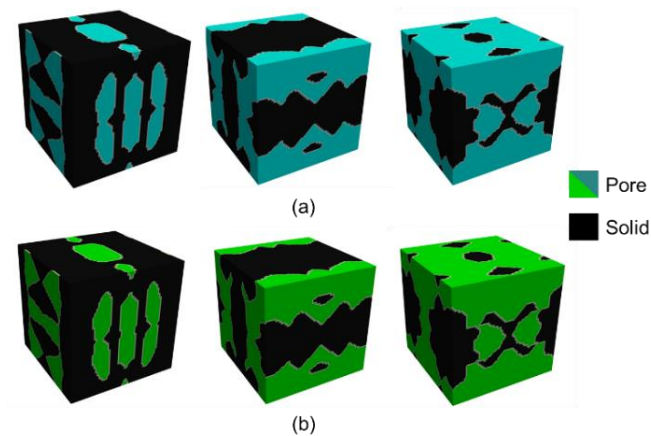


Figure 8: Reconstruction results of the proposed generative model for generating voxel images of porous metamaterials: (a) original samples in the training dataset and (b) reconstructions.

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 generative accuracy measured by the coefficient of determination (R^2) is 0.89.

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 [74] 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 LLM models much faster and more efficient in capturing longer-range dependencies [75, 76]. 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[71] nature. Nevertheless, the integration of LLMs and GNNs presented a promising method in metamaterial structural design [42, 44].

5. CONCLUSION

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 without prior knowledge of the existing connection rules. This indicates that the LLM can predict graph connections even when the connection rule is unknown, by extracting and training on skeleton graphs. Additionally, we showcase the generation of novel metamaterial samples using the proposed models.

In future works, we intend to enhance the model by utilizing strategies like Low-rank adaptation[78] and parameter-efficient fine-tuning [79, 80]. These methods aim to boost the model's accuracy and enable the use of a more complex and comprehensive model than byt5-base. 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.

REFERENCES

1. Zheng, X., et al., *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., et al., *Coulombic friction in metamaterials to dissipate mechanical energy*. Extreme Mechanics Letters, 2020. **40**: p. 100847.
4. Claeys, C., et al., *Design and validation of metamaterials for multiple structural stop bands in waveguides*. Extreme Mechanics Letters, 2017. **12**: p. 7-22.
5. Qian, J., et al., *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., et al., *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., et al. *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., et al. *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., et al., *Generative adversarial networks for the design of acoustic metamaterials*. J Acoust Soc Am, 2021. **149**(2): p. 1162.
10. Alberdi, R., et al., *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., et al., *Connectivity-guaranteed porous synthesis in free form model by persistent homology*. Computers & Graphics, 2022. **106**: p. 33-44.
17. Swartz, K.E., et al., *Manufacturing and stiffness constraints for topology optimized periodic structures*. Structural and Multidisciplinary Optimization, 2022. **65**(4): p. 129.
18. Holdstein, Y., et al. *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., et al., *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., et al., *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., et al., *Descriptor-based methodology for statistical characterization and 3D reconstruction of microstructural materials*. Computational Materials Science, 2014. **85**: p. 206-216.
23. Meyer, P.P., et al., *Graph-based metamaterials: Deep learning of structure-property relations*. Materials & Design, 2022. **223**: p. 111175.
24. Makatura, L., et al., *Procedural metamaterials: A unified procedural graph for metamaterial design*. ACM Transactions on Graphics, 2023. **42**(5): p. 1-19.
25. Yamaguchi, K., et al., *Graph-theoretic estimation of reconfigurability in origami-based metamaterials*. Materials & Design, 2022. **213**: p. 110343.
26. Du, P., et al., *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., et al., *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., et al., *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., et al., *The graph neural network model*. IEEE transactions on neural networks, 2008. **20**(1): p. 61-80.
36. Wu, Z., et al., *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., et al. *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., et al., *Large language models on graphs: A comprehensive survey*. arXiv preprint arXiv:2312.02783, 2023.
43. Vaswani, A., et al., *Attention is all you need*. Advances in neural information processing systems, 2017. **30**.
44. Li, Y., et al., *A survey of graph meets large language model: Progress and future directions*. arXiv preprint arXiv:2311.12399, 2023.
45. Xie, H., et al., *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., et al., *Graph-based modeling of online communities for fake news detection*. arXiv preprint arXiv:2008.06274, 2020.
48. Zhao, H., et al., *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., et al. *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., et al. *Fine-tuning multimodal llms to follow zero-shot demonstrative instructions*. in The Twelfth International Conference on Learning Representations. 2023.

53. Perozzi, B., et al., *Let Your Graph Do the Talking: Encoding Structured Data for LLMs*. arXiv preprint arXiv:2402.05862, 2024.
54. He, X., et al. *Harnessing explanations: Llm-to-lm interpreter for enhanced text-attributed graph representation learning*. in *The Twelfth International Conference on Learning Representations*. 2023.
55. Thomas, N.K. and M. Welling, *Variational graph auto-encoders*. arXiv preprint arXiv:1611.07308, 2016. **2**(10).
56. Pan, S., et al., *Adversarially regularized graph autoencoder for graph embedding*. arXiv preprint arXiv:1802.04407, 2018.
57. Wang, C., et al., *Attributed graph clustering: A deep attentional embedding approach*. arXiv preprint arXiv:1906.06532, 2019.
58. Sun, D., et al., *Dual-decoder graph autoencoder for unsupervised graph representation learning*. Knowledge-Based Systems, 2021. **234**: p. 107564.
59. Kingma, D.P. and M. Welling, *Auto-encoding variational bayes*. arXiv preprint arXiv:1312.6114, 2013.
60. Xu, K., et al., *Graph2seq: Graph to sequence learning with attention-based neural networks*. arXiv preprint arXiv:1804.00823, 2018.
61. Sarkar, S. and L. Lausen, *Testing the limits of unified seq2seq LLM pretraining on diverse table data tasks*. 2023.
62. Pu, Y., *fine-tuned llm as program writers*, in <https://evanthebounicy.github.io/program-synthesis-minimal/generation-with-llm/>. 2022.
63. Zhang, H., et al., *A survey of controllable text generation using transformer-based pre-trained language models*. ACM Computing Surveys, 2023. **56**(3): p. 1-37.
64. Raffel, C., et al., *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.
65. Fu, Z., et al., *Decoder-Only or Encoder-Decoder? Interpreting Language Model as a Regularized Encoder-Decoder*. arXiv preprint arXiv:2304.04052, 2023.
66. Pourpanah, F., et al., *A review of generalized zero-shot learning methods*. IEEE transactions on pattern analysis and machine intelligence, 2022.
67. Xue, L., et al., *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.
68. Shen, Z., et al. *Efficient attention: Attention with linear complexities*. in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*. 2021.
69. Szabo, F., *The linear algebra survival guide: illustrated with Mathematica*. 2015: Academic Press.
70. Otair, D.M., *Approximate k-nearest neighbour based spatial clustering using kd tree*. arXiv preprint arXiv:1303.1951, 2013.
71. Ouyang, S., et al., *LLM is Like a Box of Chocolates: the Non-determinism of ChatGPT in Code Generation*. arXiv preprint arXiv:2308.02828, 2023.
72. Pu, Y., et al., *Program synthesis with pragmatic communication*. Advances in Neural Information Processing Systems, 2020. **33**: p. 13249-13259.
73. Pu, Y., et al., *Learning to select examples for program synthesis*. 2018.
74. Zhang, C., et al., *A simple llm framework for long-range video question-answering*. arXiv preprint arXiv:2312.17235, 2023.
75. Raiaan, M.A.K., et al., *A review on large Language Models: Architectures, applications, taxonomies, open issues and challenges*. IEEE Access, 2024.
76. Liu, X., et al. *Learning to encode position for transformer with continuous dynamical model*. in *International conference on machine learning*. 2020. PMLR.
77. Bhattacharjee, A., et al., *Towards LLM-guided Causal Explainability for Black-box Text Classifiers*. 2024.
78. Hu, E.J., et al., *Lora: Low-rank adaptation of large language models*. arXiv preprint arXiv:2106.09685, 2021.
79. Liu, H., et al., *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.
80. Mo, Y., J. Yoo, and S. Kang, *Parameter-Efficient Fine-Tuning Method for Task-Oriented Dialogue Systems*. Mathematics, 2023. **11**(14): p. 3048.