buildings

MDPI

Article

Generation Method for HVAC Systems Design Schemes in Office Buildings Based on Deep Graph Generative Models

Hongxin Wang [1], Ruiying Jin [1], Peng Xu [1,*] and Jiefan Gu [2]

[1] School of Mechanical Engineering, Tongji University, Shanghai 201804, China; hongxin_wang@tongji.edu.cn (H.W.); 2211118@tongji.edu.cn (R.J.)
[2] College of Architecture and Urban Planning, Tongji University, Shanghai 200092, China; gu_lavender@outlook.com (J.G.)
* Correspondence: xupeng@tongji.edu.cn

Abstract: The design process of heating, ventilation, and air conditioning (HVAC) systems is complex and time consuming due to the need to follow design codes. Since the design standards are not fixed, the final outcome often depends on the designer's experience. The development of building information modeling (BIM) technology has made information throughout the building lifecycle more integrated. BIM-based forward design is now widely used, providing a data foundation for combining HVAC system design with machine learning. This paper proposes an unsupervised learning method based on deep graph generative models to uncover hidden design patterns and optimization strategies from the design results. We trained and validated four deep graph generative models—GAE, GNF, GAN, and diffusion—using HVAC system terminal pipeline layout data. Accuracy and precision metrics were used to compare the generated designs with automated forward design solutions, assessing the models' ability to capture both local variations and broader changes in design logic. A graph-neural-network-based evaluation method was employed to measure the models' capacity to detect changes. The results indicate that all four models achieved prediction accuracies exceeding 90% and precision rates above 75%. The models effectively captured both local modifications made by designers and global design changes, showing greater sensitivity to global layout adjustments than to local updates. When comparing the results generated by deep graph generative models and the actual design, it is obvious that the accuracy of the predictions varies significantly due to the complexity of the test buildings.

Keywords: building information modeling; HVAC system design; generative design; machine learning; deep graph generative models

Citation: Wang, H.; Jin, R.; Xu, P.; Gu, J. Generation Method for HVAC Systems Design Schemes in Office Buildings Based on Deep Graph Generative Models. *Buildings* **2024**, *14*, 3405. https://doi.org/10.3390/buildings14113405

Academic Editor: Ricardo M. S. F. Almeida

Received: 16 September 2024
Revised: 9 October 2024
Accepted: 23 October 2024
Published: 26 October 2024

1. Introduction

Building information modeling (BIM) has been widely recognized as a significant breakthrough in the architecture, engineering, construction, and owner-operated (AECO) sector [1]. Unlike traditional two-dimensional computer-aided design (CAD) drawings, BIM offers unified, parametric, and visual models, enabling direct design in three dimensions [2]. This shift provides a solid foundation for automating and optimizing design processes. Heating, ventilation, and air conditioning (HVAC) systems, as an important component of building energy systems, are often constrained in the design process by the engineers' experience, capabilities, and time limitations [3]. Therefore, BIM provides a standardized data foundation that enables the integration of automated and intelligent methods in HVAC system design practices [4].

Generative design is a process that uses computer algorithms to automatically generate design outcomes based on parametric inputs, allowing designers to automate parts of the design process [5]. With the advancements in machine learning, this approach has unlocked new opportunities, enabling the development of parametric models specifically

for automated design in HVAC systems [6]. In practical system design, human designers often intervene in design results generated by traditional rule-based methods, which highlights the limitations of the rule-based methods that fail to account for the nuances of professional design habits. In contrast, parametric machine learning approaches can recognize and adapt to these subtleties, leading to design outcomes that more closely mirror human-generated solutions.

This study focuses on the layout design of terminal pipelines within HVAC hydronic systems, where deep graph generative models are trained and validated. It introduces accuracy and precision metrics to assess the generative designs, comparing them against automated design solutions. This study evaluates the model's ability to capture both local variations and shifts in overall design logic. Additionally, it incorporates a previously proposed graph-neural-network-based evaluation method to demonstrate the deep generative model's capability to extract implicit design habits and optimization strategies used by designers, as observed in case studies. Finally, this study investigates whether the graph neural network evaluation model can effectively differentiate the criteria designers use to assess and select design schemes.

2. Literature Review

2.1. Generative Design in HVAC

Generative design employs optimization methods to target objectives such as building energy consumption, economic efficiency, or occupant comfort, ultimately selecting appropriate design parameters to achieve optimal architectural solutions [7]. At its core, it involves generators and evaluators, using an iterative process of generation and evaluation to create novel design solutions [8]. This approach can creatively produce a variety of complex solutions and is commonly applied in fields such as art, engineering, and product design, and performance-based design is an example of generative design in the architectural domain.

Generative design techniques emulate the principles of natural evolution in the design process and are commonly categorized into the following four types of algorithms in architectural design: shape grammars (SG), Lindenmayer Systems (LS), cellular automata (CA), and genetic algorithms (GA). Shape grammars involve a set of rules that iteratively apply geometric transformations to generate and modify designs [9]. For instance, Stiny and Mitchell [10] utilized parametric shape grammar to create a floor plan for Palladian villas, while Downing and Flemming [11] applied it to represent spatial organization rules for single-story houses. Lindenmayer Systems, or L-systems, are generative rules applied recursively to strings. Unlike shape grammars, they operate on auxiliary conditions rather than directly on shapes and have been used in generating road networks and for constructing shapes [12]. Cellular automata are grids of cells on specific shapes, evolving based on rules driven by the states of neighboring cells [13]. Genetic algorithms are heuristic algorithms inspired by the process of natural selection, widely used for generating high-quality solutions to optimization problems, often described using variables, constraints, and objective functions.

Among these four algorithms, architects and HVAC engineers primarily utilize genetic algorithms for design optimization. In the HVAC field, genetic algorithms are employed to optimize various building design parameters, including building orientation, shape, window-to-wall ratio, shading [14–16], building envelope properties (such as the thermal performance of walls, roofs, and windows) [17,18], HVAC system capacity configurations [19,20], and renewable energy integration [17]. Tuhus-Dubrow and Krarti [21] developed a simulation optimization tool using genetic algorithms to optimize building shapes and envelope features. Palonen et al. [22] solved a single-objective optimization problem using genetic algorithms to minimize the life cycle cost of standalone residences, optimizing parameters such as insulation thickness, window U-values, and heat recovery type. Wright and Farmani [23] synchronized the optimization of building structure,

HVAC system sizing, and control strategies for individual HVAC zones using genetic algorithms. Asiedu et al. [24] designed HVAC systems with minimal lifecycle costs using genetic algorithms. Furthermore, Berquist et al. [25] proposed a methodology for employing genetic algorithms throughout various stages of the HVAC design process, developing a MATLAB program to generate zoning strategies for given floor plans.

Generative design algorithms facilitate the exploration of vast potential design spaces, enabling automated evaluation and optimization of these solutions [26]. They extract and transform design features through induction and reasoning, thereby automating the design process. Generative design can parallelize design tasks, manage extensive information efficiently, and assist designers in identifying optimal design forms through automatic feedback and simulation results.

2.2. Deep Generative Models

With the advancement of traditional deep learning techniques such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs), substantial success has been achieved in the mining of Euclidean space data, such as images, as well as sequence data, including text and signals. However, graphs, as a unique form of non-Euclidean data that encapsulate the complex relationships between nodes and edges, cannot be directly learned using traditional deep learning methods [27]. This has led to the emergence of graph neural networks (GNNs). GNNs are a neural network architecture specifically designed for training on graph-structured data. They iteratively update the representation of nodes by aggregating the representations of neighboring nodes and their feature representations from the previous iteration.

GNNs can be categorized according to their downstream tasks into graph-level regression/classification tasks, node-level regression/classification tasks, and edge-level regression/classification tasks. From the perspective of graph topology, GNNs can also be classified into static and dynamic graph structures [28]. In static graph structures, the topology remains unchanged after formation, while dynamic graphs change nodes or edges. Recently, GNN models have seen widespread application in the HVAC domain. Table 1 summarizes the research tasks related to GNN applications, detailing the graph data structures and GNN methods used for each type of application.

Table 1. Research on the application of graph neural networks in HVAC.

Application	Grap Structure	Methods	Reference
Fault diagnosis of HVAC	Correlation graphs of building operation data	GCN	Fan et al. [29]
Building energy simulation	Parameter correlation graphs based on empirical knowledge	GraphSAGE	Chen et al. [30]
Building load prediction	Graphs of building spatial relationships with building load results	GCN	Lu et al. [31]
Regional building energy simulation	Correlation graphs of regional building energy consumption data	ST-GCN	Hu et al. [32]
Human thermal comfort	Heterogeneous data relationship graph for different entity data	GNNs	Gnecco et al. [33]
Building occupancy prediction	Spatial relationship graph of the building with occupancy	GNNs-LSTM	Xie et al. [34]
System control optimization	Device correlation graph with spatial and temporal data	GNN-RNN	Zhang et al. [35]

Graph neural networks (GNNs) are capable of learning not only explicit design rules, but also implicit rules derived from existing designs, enabling the generation of more complex alternative designs. As an end-to-end learning model, GNNs allow raw data to be directly input into the model without the need for manual feature extraction [36]. Furthermore, GNNs can automatically learn design rules, thereby reducing the demand for

programming expertise. Currently, the use of GNNs in generative design is largely limited to floor plan generation, indicating that the full potential of GNNs in generative design has not yet been fully realized. While GNN-based generative design has been extensively applied in other domains, such as structural design, gas distribution network planning, and urban road traffic design, there remains a lack of sufficient research on its application in the design of HVAC systems.

2.3. Deep Graph Generative Models

Deep generative models, a subset of generative models based on deep learning techniques, aim to generate new data samples by learning high-dimensional representations of data distributions [26]. These models typically rely on neural network architecture and are trained using the backpropagation algorithm. By modeling the probability density of observable data, deep generative models can stochastically generate sample data. The incorporation of multiple hidden layers into these models enhances their complexity and learning capacity, allowing them to produce more sophisticated results. Currently, deep generative models find extensive applications across various domains, such as natural language processing, speech recognition, and computer vision.

Deep generative models have demonstrated a remarkable capacity to learn complex data distributions, enabling them to produce high-quality, realistic data samples such as photorealistic images and coherent natural language sentences. Many of these models employ unsupervised learning techniques, allowing them to infer data distributions without requiring additional labels or supervision [37]. This capability empowers deep generative models to generate a diverse array of data samples, extending beyond the predominant patterns present in the training data. Moreover, these models often learn latent space representation, which facilitates interpolation operations that yield smooth transitions between data points. For instance, in image generation tasks, this can result in a sequence of gradually evolving images. Some deep generative models also offer a degree of interpretability, with their outputs being explicable in terms of variations along different dimensions of the latent space.

Deep graph generative models represent a sophisticated class of generative models specifically designed for creating graph-structured data, distinct from traditional generative models that typically focus on linear data types. These models are adept at generating data that inherently possess graph structures, such as social networks, molecular structures, and transportation networks [38]. The primary challenge for deep graph generative models lies in learning how to represent the graph structure, which involves capturing the characteristics of nodes and edges, as well as the relationships between them. Common approaches for this representation include encoding nodes as vectors, node embeddings, or subgraph embeddings [39].

In this study, four deep graph generative models are investigated, which are graph autoencoders [40], generative adversarial networks [41], diffusion neural network models [42], and normalized flow models [43].

Autoencoders, GANs, normalizing flow models, and diffusion neural network models are traditionally employed for modeling and training data in Euclidean spaces. However, graph-structured data reside in non-Euclidean spaces. Consequently, applying these models to graph data in this research necessitates mapping graph-structured data to Euclidean spaces through graph embedding techniques.

Current research on graph generation often assumes that graphs are homogeneous, however, different subgraphs within a larger graph may exhibit diverse structural distributions [44]. Real-world graphs encode semantic information through node and edge labels, necessitating that graph generative models learn both the structural and label information concurrently. Many existing models focus primarily on learning graph structures without adequately capturing the labels of nodes and edges, which often encode critical semantic information and influence the graph's structural properties.

While some models are capable of generating domain-specific graphs under specific constraints, they frequently lack the generalizability required to apply to other domains. Experimental studies in this area are often not comprehensive, relying on simplistic and generalized evaluation metrics or focusing predominantly on synthetic or small datasets. For deep graph generative models to be truly effective, they must be capable of scaling up to larger datasets, thus harnessing more extensive information to generate more realistic and representative graphs.

3. Research Design and Framework

The technical route of this study is shown in Figure 1, in which the building information model of an actual building is first utilized to conduct a forward design of the HVAC system using conventional forward design methods in order to obtain the forward design results for the building's HVAC system. Subsequently, we generate graph-structured data suitable for deep graph neural network model predictions by employing methods that construct the geometric and topological data of the building space, as well as the topology of the HVAC system. Based on the deep graph neural networks and deep graph generative models, we perform a predictive analysis on the aforementioned graph-structured data, generating corresponding HVAC system design solutions and evaluating their performance. Finally, we compare the results generated by the deep generative model with the actual design outcomes of the building and the forward design results.

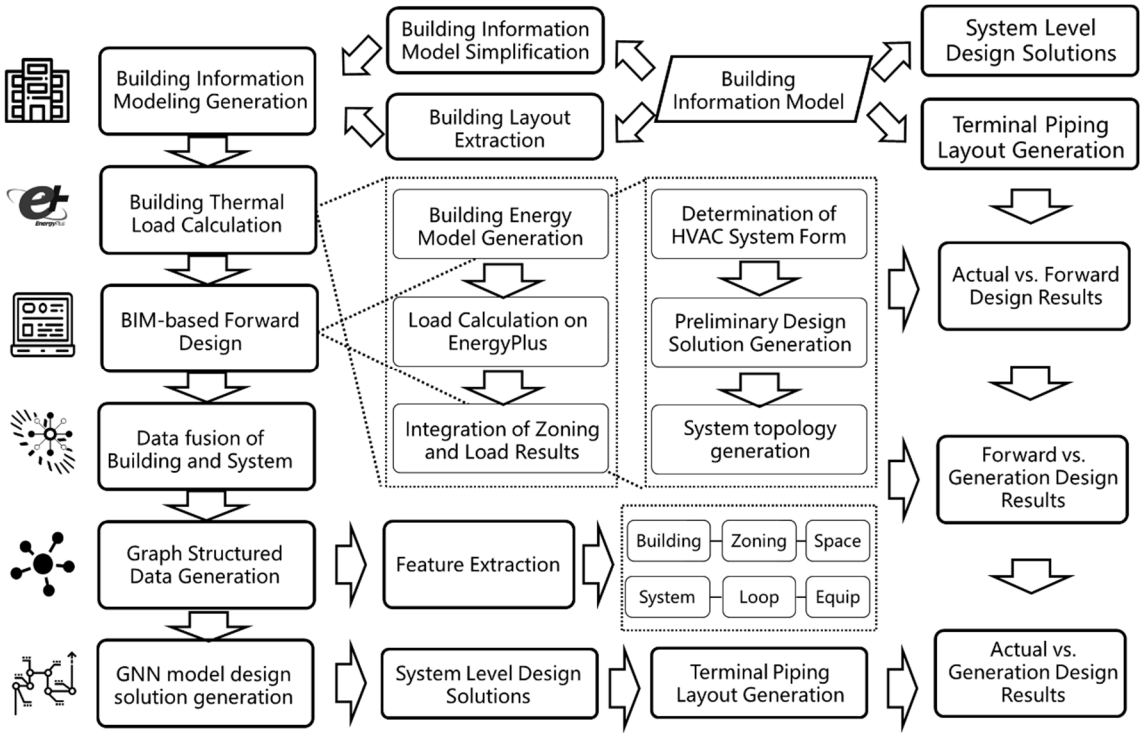


Figure 1. Technical route of this study.

4. Methodology

The focus of this study is on the piping layout solutions for the terminal water systems in HVAC systems. Therefore, we model the terminal water piping system as graph-structured data, where nodes represent terminal devices and edges represent the pipes connecting these devices. The feature values of terminal devices are defined as node features, while the feature values of pipes are defined as edge features. The objective of applying the GNN model is to generate a corresponding graph-structured data representation of the HVAC terminal water system's piping topology based on the existing building

models and their spatial distribution as graph-structured data. Given the above setup, the research problem for this section is described as follows:

As a notational convention, a graph is generally represented as $G = (V, E)$, where $V = \{v_1, \cdots, v_n\}$ is the set of nodes and $E = \{(v_i,\ v_j)|v_i,\ v_j \in V\}$ is the set of edges. Both nodes and edges in a graph may have labels, which can be represented as mappings $\mathcal{L}_n: V \to \mathbb{V}$ and $\mathcal{L}_e: E \to \mathbb{E}$, where $\mathcal{L}_{\mathrm{n}}$ and $\mathcal{L}_{\mathrm{e}}$ are sets of node and edge labels, respectively. The labels of a node v and an edge e are denoted by $L(v)$ and $L(e)$.

In this study, we assume that all graphs are connected and contain no self-loops. The objective of a graph generative model is to learn a distribution $P_{model}(\mathbb{G})$ from a given set of observed graphs $\mathbb{G} = \{G_1, \cdots G_m\}$, where this set of graphs is derived from an underlying hidden distribution $P(\mathbb{G})$. Each graph G_i may have a different number of nodes and edges, as well as a different number of node and edge labels.

A graph generative model is considered effective when the learned distribution of graphs closely approximates the hidden distribution of the graphs, i.e., $P_{model}(\mathbb{G}) \approx P(\mathbb{G})$. In summary, a graph generative model must be capable of generating graphs similar to those produced by the distribution $P(\mathbb{G})$ without any prior assumptions related to the structure or labels of the graphs.

This study leverages the architectural and HVAC system topological datasets from prior research, extracting both the architectural spatial hierarchy topology model and the system equipment hierarchy topology model as inputs for the generative model employed in this section. Through the training of the encoder and decoder black-box models, a generative model capable of producing diverse terminal design schemes for HVAC systems is ultimately obtained. As illustrated in Figure 1, the encoder in the model presented in this section consists of multiple hidden layers, including message-passing layers, pooling layers, and linear layers. After processing the graph data through the encoder of the generative model, the output is transformed into low-dimensional space encoding, which is then restored to a graph structure identical to that of the input data structure via the model's decoder. Depending on the decoding method, decoders are classified into function-decoding white-box decoders and GNN-based black-box decoders. Furthermore, the generative model can be categorized into different forms based on the specific loss function employed during model training.

4.1. System-Level Data Generation

The generation of system-level designs is based on the automated design methodology for HVAC systems [45]. This encompasses two key aspects of design, as follows: (1) the selection of heating and cooling sources for HVAC systems and the generation of topological connections according to the total cooling and heating load of the building; and (2) the selection of terminal equipment for the HVAC water system based on the zoning results and the distribution of loads in the building plan. Since this study focuses on HVAC systems utilizing chillers and boilers as heating and cooling sources and fan coil units as terminal equipment, the discussion is limited to the topology and equipment selection specific to this system configuration.

The layout of terminal piping in HVAC water systems is the primary focus of this research. The topological connection diagram of terminal equipment serves as the graph-structured data at the equipment level within the database for describing the HVAC system graph data structure. This study employs two methods, the minimum spanning tree and the minimum Steiner tree, under four different boundary conditions, to generate eight categories of samples. The specific methodology for generating these designs will not be elaborated here. These eight types of samples can simulate the iterative process of design modifications by designers during the schematic design phase and, therefore, can be used to assess the sensitivity of the deep graph generative models developed in this study to both global and local design changes made by designers.

This study employs two building model generation approaches—survey-based and cellular-automata-based—to generate floor layouts. As shown in Figure 2, a total of 550

building layouts with varying spatial structures were created to serve as input conditions for the generation of HVAC system designs. Based on the generated building layouts and pre-defined design parameters, such as the number of floors, floor height, room dimensions, and window-to-wall ratio, a sample space comprising a large number of building model samples was constructed. Through random sampling from this sample space, 2000 building models were selected for the generation of topological graph data for both the buildings and the HVAC systems. Using these generated building models, in conjunction with automated HVAC system design methods, this study employed a parametric approach to generate HVAC system data, including load calculations, system selection, and the connection of terminal equipment. The generated cases were stored in the graph database Neo4j in the form of graph data, facilitating easy access to and modification of the data.

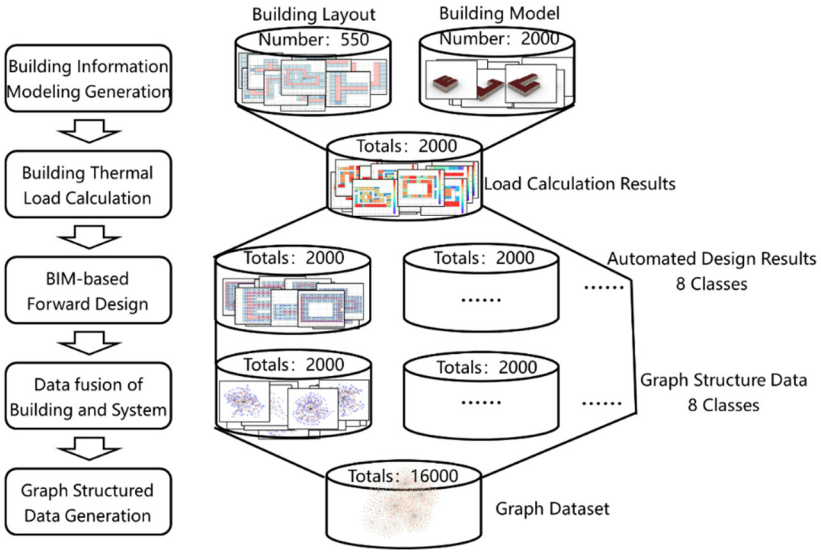


Figure 2. The process of generating the case dataset.

4.2. Architecture of the Deep Graph Generative Model

Based on the aforementioned methodology, this study implements four distinct structures of deep graph generative models, each differing in their encoder/decoder architecture and training loss functions. These models are trained on the previously mentioned dataset, and their training processes are depicted in Figure 3.

The primary objective of the aforementioned models is to learn the probability distribution of data and generate new data samples, allowing for their comparison and substitution to some extent. Autoencoders, diffusion models, and flow models can be used for probability density estimation by learning the data's probability distribution. Each model employs distinct methods to model data distribution, as follows: autoencoders minimize reconstruction error, diffusion models iteratively generate data through diffusion processes, and flow models use invertible transformations to map simple distributions to complex ones. GANs, diffusion models, and flow models focus on generating new data samples that resemble the true data distribution. They employ various mechanisms for data generation, as follows: autoencoders produce predictive data through the encoding and decoding of input sample vectors, GANs create realistic data via adversarial training, diffusion models generate data through iterative diffusion processes, and flow models produce complex distributions from simple ones through invertible transformations.

All four deep graph generative models utilize graph-structured data as the input. They employ various encoding techniques to map this data into a low-dimensional latent space. During the prediction phase, the models use different decoders to reconstruct the low-dimensional latent data back into high-dimensional, graph-structured data. This

enables the generation of graph-structured data through the models' encoding and decoding mechanisms.

Within this graph generation framework, the four models differ in their encoding/decoding approaches and loss function calculations. These models can be broadly categorized into the following two types: (1) autoencoders and GANs, which leverage GNNs with message-passing layers as the primary structure for both encoders and decoders; and (2) normalizing flow models and diffusion models, which utilize functional mappings to encode and decode graph-embedded data.

The key distinction between autoencoders and GANs lies in their training methodologies. Autoencoders employ conventional loss functions to compute loss values, whereas GANs use a discriminator to determine the generator's training direction based on the similarity between generated and real data. Normalizing flow models differ from diffusion models in that they rely on reversible functional mappings to transform graph-embedded latent variables into low-dimensional spaces and then use inverse transformations to decode these variables. In contrast, diffusion models establish a noise-based framework, mapping graph-embedded latent variables into a stochastic variable space through noise addition and removal, and subsequently reconstructing the original data from this space.

This research focuses on the hierarchical topological connections of terminal devices in HVAC systems, which are represented as homogenous graphs with attributed nodes and edges. The graphs used in this study are undirected, particularly those depicting the water supply topology of terminal systems, while omitting return water pipelines, thereby classifying them as undirected attributed graphs. These graph data types are employed in training the four deep graph generative models.

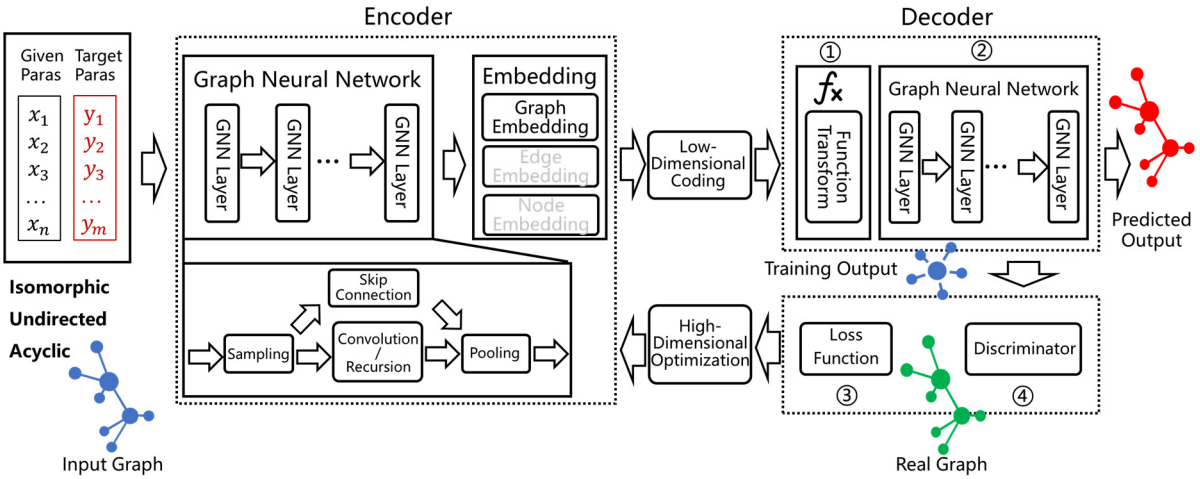


Figure 3. Training process of deep graph generative models.

To meet this study's objectives of generating system designs and terminal layout solutions from architectural topology graph data, both model inputs and outputs have been tailored accordingly. As shown in Figure 4, for autoencoders and GANs, architectural topology graphs and random noise graphs generated from these topology graphs serve as the inputs, while the output is the system topology graph generated by the model. The random noise graph acts as a foundational basis for system topology graph generation, with training losses calculated in reference to the actual system topology graph. Normalizing flow models and diffusion models both employ architectural topology graphs and random noise graphs as inputs, producing architectural and system topology graphs as outputs. During training, the generated system topology graph is compared with the real system topology graph to calculate loss, which is then backpropagated through graph embeddings and function mapping/noise models to update the model parameters.

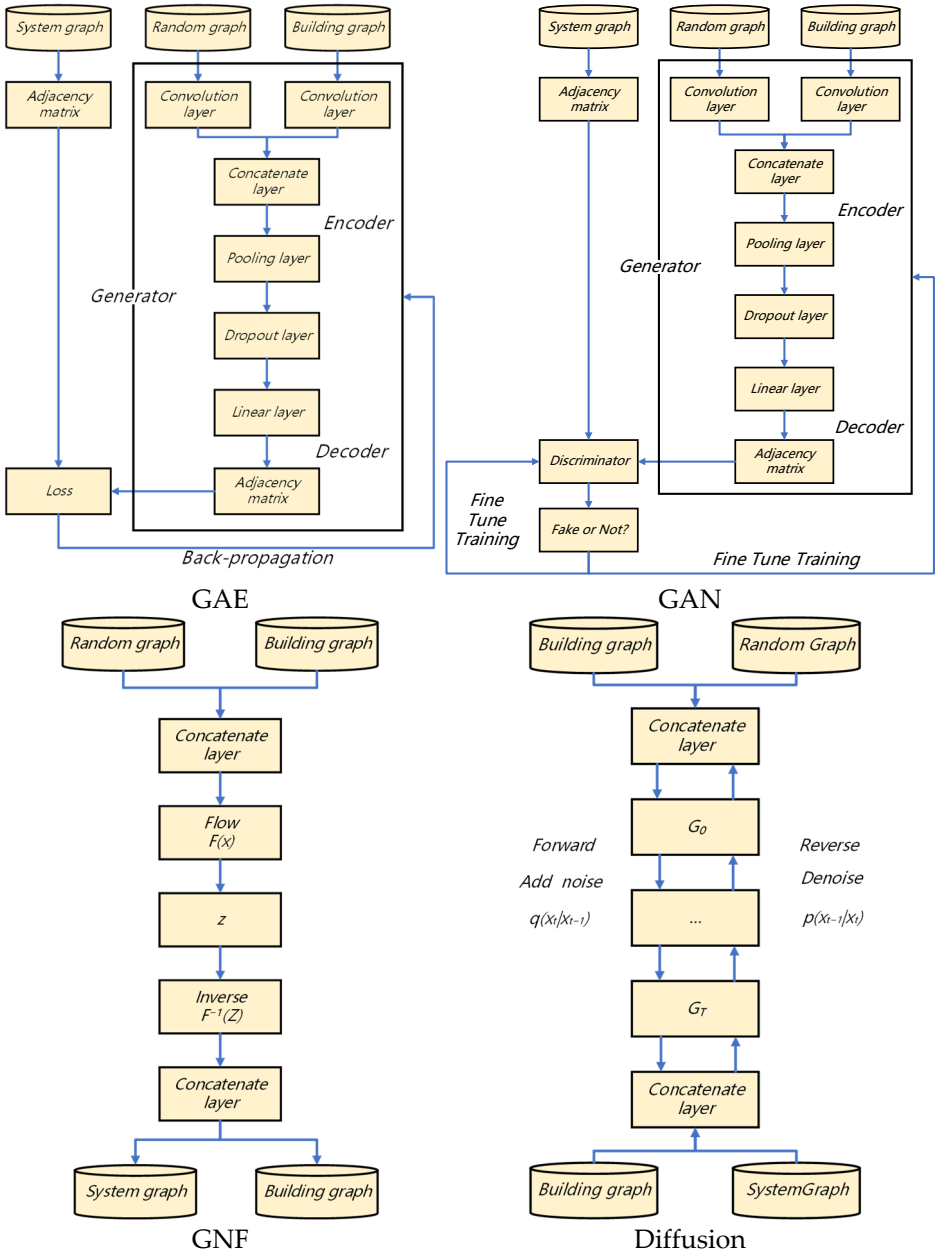


Figure 4. Deep graph generative models' structure.

4.3. Deep Graph Generative Model Training

This study employs the following four distinct generative models to train on the previously generated case studies: GAE, GAN, diffusion neural network model (diffusion), and normalizing flow model (GNF). Based on the encoder architecture, two additional variants were derived: the variational graph autoencoder (VGAE) and the variational normalizing flow model (VGNF). Therefore, the deep graph generative models selected for this research total six, as follows: GAE, VGAE, GAN, diffusion, GNF, and VGNF. In the following sections, these models will be referred to by their respective acronyms. A brief overview of the structure of the six deep graph generative models used in this study is provided below.

(1) GAE and VGAE

The encoder component of the GAE comprises four convolutional layers and one linear layer, with ReLU-based activation layers placed between every two convolutional layers. In contrast, the encoder of the VGAE includes an additional convolutional layer to

compute the mean μ and standard deviation σ for the variational inference process. Both the GAE and VGAE employ the same decoder architecture, which generates the adjacency matrix of the output graph by computing the inner product of the encoder's output matrix and its transpose.

(2) GNF and VGNF

The encoder of the GNF is composed of four convolutional layers and one linear layer, with ReLU-based activation functions applied between every pair of convolutional layers. Unlike the GAE, the GNF encoder includes an additional forward computation layer consisting of two convolutional layers, a sampling layer, and two backward computation layers. The VGNF encoder extends the GNF encoder by incorporating an extra convolutional layer to compute the mean (μ) and standard deviation (σ) required for variational inference. The decoders of both the GNF and the VGNF mirror those of the GAE and VGAE, generating the graph's adjacency matrix by calculating the inner product of the encoder's output matrix and its transpose.

(3) GAN

The generator of the GAN comprises two convolutional layers followed by four linear layers, with ReLU activation functions applied between each pair of convolutional and linear layers. The output layer of the generator consists of two convolutional layers, configured identically to the encoder of the VGAE. The discriminator in the GAN is similarly structured with two convolutional layers and four linear layers, interspersed with ReLU activation functions between each pair of convolutional and linear layers. Additionally, dropout layers are integrated within the convolutional layers of the discriminator to enhance model robustness.

(4) Diffusion

The diffusion model's forward computation process consists of four convolutional layers, two normalization layers, and one linear layer, with ReLU activation functions applied between each pair of convolutional layers. The backward computation process is similarly composed of four convolutional layers, two normalization layers, and two linear layers. However, the backward computation process differs in that the input includes one sampling layer and the output includes one self-attention layer. The output layer of the model also comprises two convolutional layers.

In this study, a dataset with eight different sample categories was generated, each containing 2000 graph-structured data samples. For the training of deep graph generative models, each sample category's data was split into training, testing, and validation sets, with 1600 samples allocated to the training set, 200 samples to the testing set, and 200 samples to the validation set. The models were implemented using the PyTorch framework and accelerated using GPU resources. After multiple tests to ensure convergence of all model training parameters, the final number of training steps was set to 200. The training process recorded the loss function results on both the training and testing sets for each step. To enhance the prediction accuracy and reduce the prediction loss of the deep graph generative models, hyperparameter tuning was performed.

4.4. Hyperparameter Tuning for Models

The hyperparameter tuning process for general deep learning models typically involves considering factors such as learning rate, loss function (including its hyperparameters), batch size, dropout rate, weight decay, optimizer momentum, model depth, and convolutional kernel size. In the context of the deep graph generative models discussed in this section, the primary hyperparameters adjusted include the learning rate, batch size, and number of output channels for certain models. By using the generated dataset of building and HVAC system graph structures as input data for parameter tuning, the resulting loss function curves for the six deep graph generative models employed in this study across different sample categories are depicted in Figure 5. The lines with different

colors in Figure 5 indicate the loss values during training for different combinations of hyperparameters. Additionally, the optimal hyperparameter combinations are presented in Table 2.

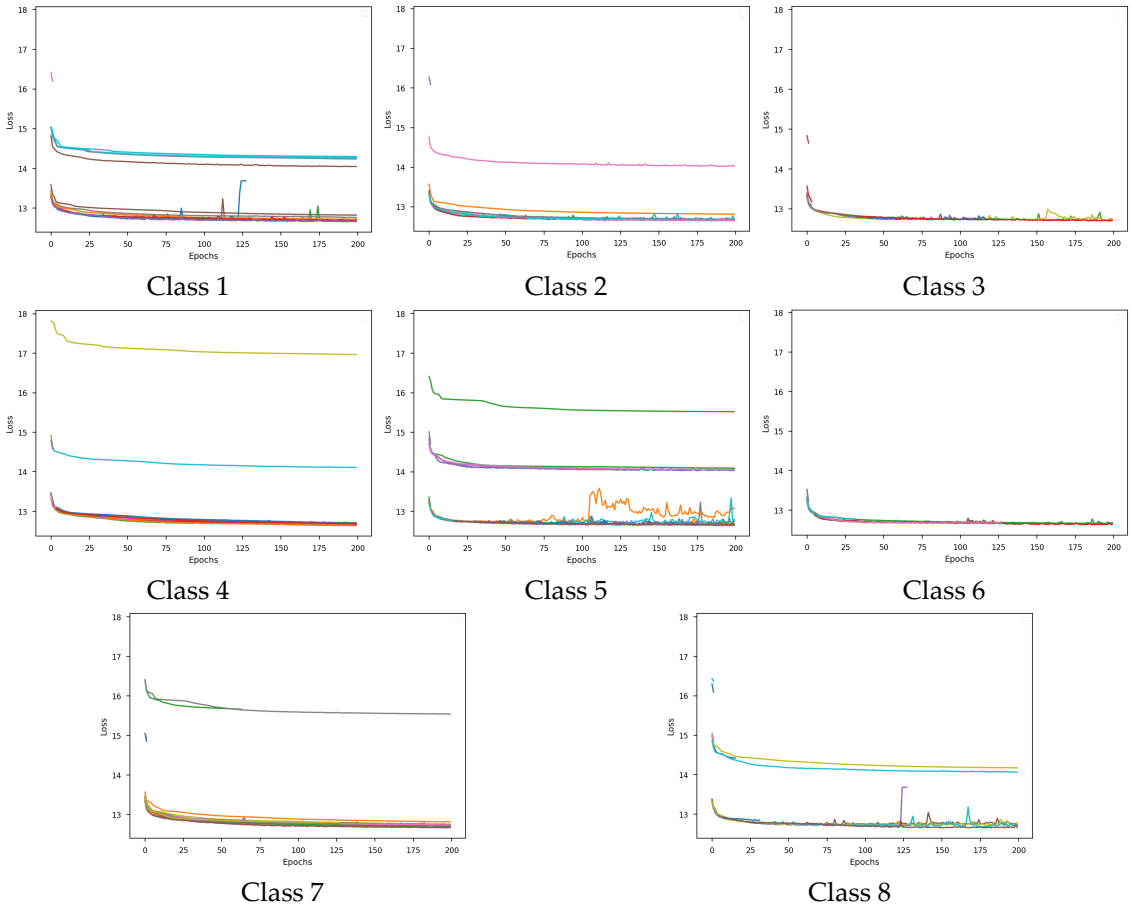


Figure 5. Loss function results of different classes in the GAE model.

Table 2. The optimal hyperparameter combination for different classes in the GAE model.

Class	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Learning ratio (10^{-4})	9.95	5.17	9.08	3.08	9.73	4.57	9.77	7.26
Batch size	8	8	8	8	8	8	8	8
Output channels	64	128	64	128	64	128	64	128

4.5. Result and Diagram Generation

This study utilizes deep graph generative models to input architectural planar topology data, augmented with random noise, enabling the model to produce diverse predicted layouts for HVAC system terminal equipment. The output of the deep graph neural network models employed in this section is represented as an adjacency probability matrix, corresponding to the graph-structured data of HVAC system terminal equipment layouts. Each element in this adjacency matrix signifies the probability of an edge existing between the adjacent nodes in the graph-structured data. In this study, a graph is represented as $G = (V, E)$, where $V = \{v_1, \cdots, v_n\}$ denotes the set of nodes and $E = \{(v_i,\ v_j)|v_i,\ v_j \in V\}$ denotes the set of edges. The adjacency probability matrix of the graph is expressed as follows:

$$A_p = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{1m} & \cdots & a_{mn} \end{bmatrix}, a_{mn} \in [0,1] \quad (1)$$

where a_{mn} denotes the probability of forming edges between the nodes output by the deep graph generative model.

In this paper, the confidence coefficient C is introduced, and the element a_{mn} in the adjacency probability matrix in Equation (1) is treated as follows:

$$a'_{mn} = \begin{cases} 0, & a_{mn} < C \\ 1, & a_{mn} \geq C \end{cases} \quad (2)$$

The adjacency probability matrix A_p is processed by Equation (2) as an adjacency matrix A whose element a'_{mn} has the value of 0 or 1. Therefore, this study transforms the problem of evaluating the generation results of the deep graph generative model into the problem of comparing the consistency of the predicted generation results with the actual results of binary classification of the elements of the adjacency matrix A'. There are many existing evaluation metrics for machine learning classification problems, among which the common ones include the following: confusion matrix, accuracy rate, error rate, precision rate, recall rate, F1 score, and Kappa coefficient. Here, the above results are compared with the original design results and calculate the accuracy and precision of the model prediction results and the actual results of the data, both of which are calculated by the following formulas:

$$Accuracy = \frac{n_{correct}}{n_{total}} = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

where $n_{correct}$ is the number of elements in the adjacency matrix A that agree with the elements at the corresponding positions of A'; n_{total} is the total number of elements in the adjacency matrix A; TP is the number of elements in the adjacency matrix A, for which the elements at the corresponding positions of A' are 1 and are predicted to be 1; TN is the number of elements in the adjacency matrix A, for which the elements at the corresponding positions of A' are 0 and are predicted to be 0; FP is the number of elements in the adjacency matrix A, for which the number of elements whose relative position elements are 0, but are predicted to be 1; and FN is the number of elements whose relative position elements are 1, but are predicted to be 0, in the adjacency matrix A'.

As shown in Equation (2), the adjacency matrix A formed by the model prediction results is related to the value of the confidence coefficient, and the larger the value of the confidence coefficient is, the fewer elements in the adjacency matrix are judged to be 1, and the fewer elements in the graph structure that can be generated as equipment connecting tubes; moreover, as shown in Equation (4), the TP of the adjacency matrix will be reduced, and the accuracy of the model prediction results will be decreased. However, the accuracy calculation result in Equation (3) is related to both TP and TN in the adjacency matrix, so it needs to be analyzed in combination with the model prediction result and the actual situation of the sample.

The above analysis is based on the similarity analysis between the model prediction results and the model sample inputs, which is not representative of an evaluation index of the merits of the model prediction results, and this study evaluates the prediction results of the model in this section in combination with the GNN-based scheme evaluation method proposed in this paper. This assessment contains four subjective and objective evaluation indexes, which are the hydraulic balance index I_H, economy index I_E, reasonableness index I_R, and aesthetics index I_A. This study will analyze the scoring

performance of the prediction results of the deep graph generative models in this section on these four indices.

5. Case Study

Three actual BIM models were selected for the generation of design solutions for HVAC systems. The feasibility and generalizability of the technical route adopted in this study will be verified by comparing the differences between the results of the parameterized design of the system based on the case generation method and the results of the automated design with the results of the actual building design.

This paper focuses on verifying the effectiveness of the deep graph network model in the HVAC terminal piping system scoring task, as well as the effectiveness of the deep graph generative model in the HVAC system solution generation task, by testing the actual building model. The BIM model of the actual building first uses the HVAC system automated design method to obtain the design results of the HVAC system. After that, combining the methods of this paper for generating the geometric topology data of the building space and the topology data of the HVAC system, we construct the graph structure data that meet the prediction of the deep graph neural network model and analyze the prediction of the above graph structure data based on the deep graph neural network and the deep graph generative model to generate the corresponding results of the HVAC system scheme and score them. At the same time, this section will also compare the results of the deep generative model with the actual design results of the building and the results of the automated design.

5.1. Building Model Simplification

Based on the BIM simplification process outlined in previous studies, this study first performs model simplification on the original building model to obtain the spatial volume model of the building. The standard layer of the original building model is picked up to construct the baseline model for the scheme generation of the deep graph generative model. As shown in Figure 6, Case 1 is characterized by a relatively complex spatial layout of the building, and the results of the terminal piping layout of the HVAC system are relatively complex, which can be used to verify the reasonableness of the terminal piping layout method of this study in the complex building plan. Case 2 is characterized by the existence of a large atrium area as a multiple connected area in the spatial layout of the building plan, which can be used to test the applicability of the terminal pipe layout method in this study for the existence of a multiple connected area. Case 3 is characterized by a larger building plan layout, a larger number of air-conditioning rooms in the plan, and a longer dry pipe stroke in the terminal piping layout of the HVAC system, which is suitable for verifying the applicability of the piping layout method of this study in the case of a larger number of terminals in the building plan layout.

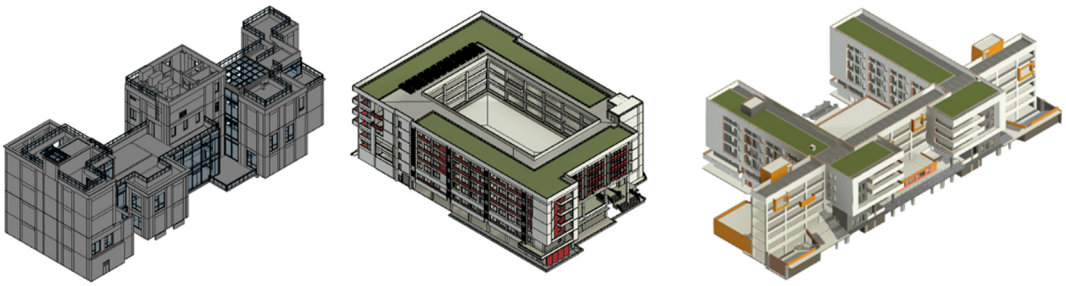


Figure 6. Building layouts of three cases.

To facilitate the processing of building information in the automated design process, this study first simplifies the original building model of the test building by eliminating

the information of beams, ramps, and other decorative structures. This simplifies the information of doors, windows, staircases, and glass curtain walls; retains only the information necessary for the conversion of the building information model into a building energy model, as shown in Figure 7; and extracts a typical building floor plan, as shown in Figure 8, as the building floor plan layout for this study in this section.

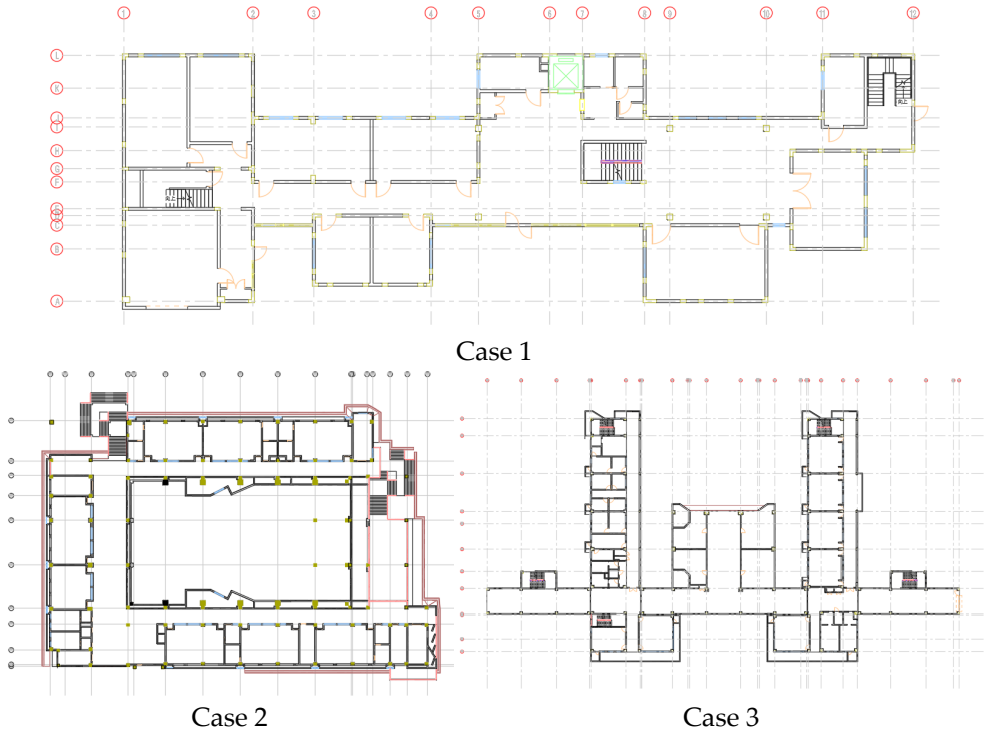


Figure 7. Building floor plans of three cases.

Figure 8. Zoning results of the building plan.

In this paper, according to the geometric layout of the building space and functional distribution of the three cases, building zoning was carried out in accordance with the scheme shown in Figure 8, in which the light blue areas are air-conditioned zones, the orange areas are corridors and open public space zones, the yellow zones are for tube wells or equipment rooms, and the gray and other white zones are non-air-conditioning zones.

The results of the actual design scheme of the terminal layout of the HVAC water system of the test case building selected in this paper are shown in Figure 9. In order to allow the actual building design results to be compared with the results of the forward design and the deep graph generative models in this paper under the same data framework, this study simplifies the actual design scheme of the case building and maps the actual design scheme to the topology constructed in this paper.

Firstly, this study simplifies the terminal equipment of the water system in the layout of the actual design scheme into graph structure data nodes, and its spatial geometric location information is matched with the results of the building plan grid division and room functional partitioning, so that we can obtain the distribution of the simplified terminal equipment nodes of the system in the building plan space.

Secondly, based on the distribution of the terminal equipment nodes of the system mentioned above, this study abstracts the piping layout results in the actual design results, obtains the piping layout scheme that can match the terminal equipment nodes, and calculates and extracts the characteristic parameters of the equipment nodes and the design parameters of the intermediate nodes on the piping layout in this layout scheme. At the same time, the results of the load calculation of the building, the results of the selection of the system cooling and heating sources, etc., are also transformed and stored in this study.

Finally, this study maps the actual design schemes of the three selected case buildings to the topology data constructed in this paper, and the results of the terminal piping layout are shown in Figure 10.

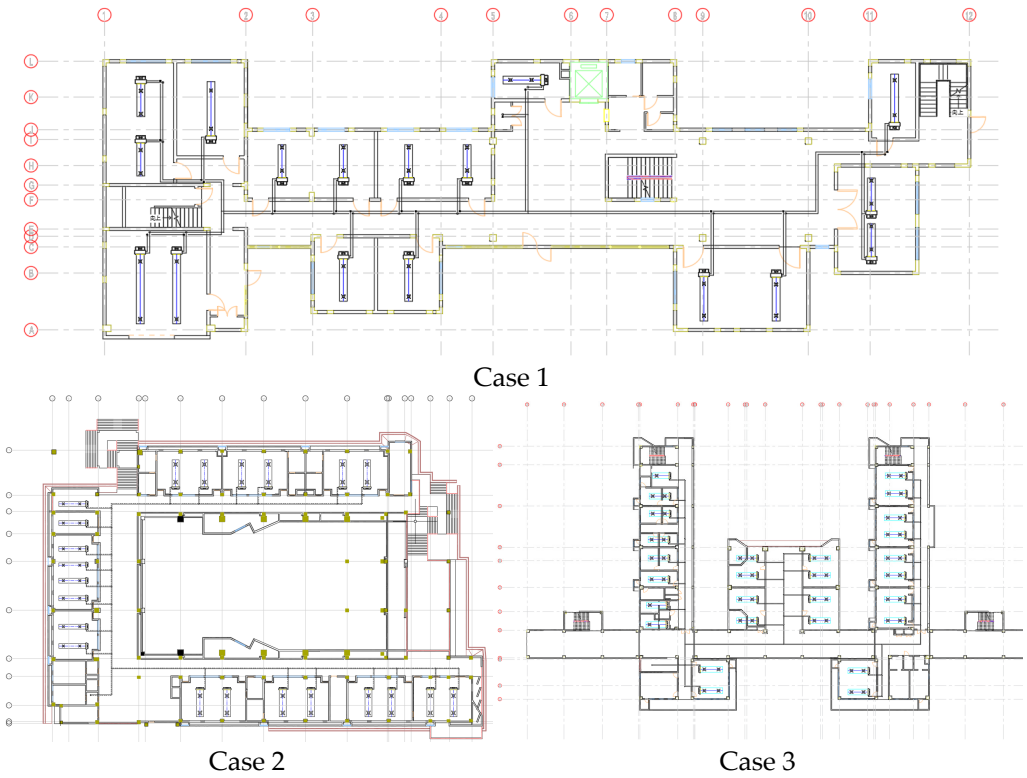
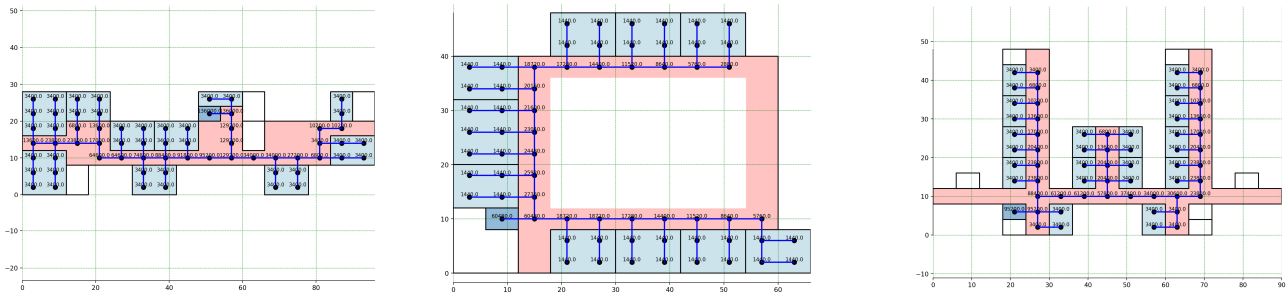


Figure 9. Results of the actual design plan for the terminal layout of the HVAC water system in case buildings.

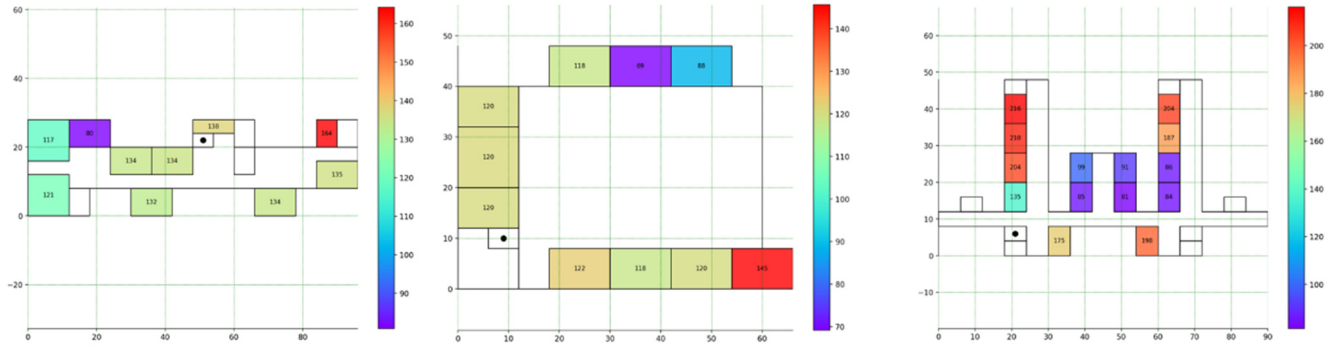


Case 1 Case 2 Case 3

Figure 10. Case building system actual design solution conversion results.

5.2. Automated Design Solution Generation

The above simplified building model is used to form a building energy model for load calculation by setting the building envelope parameters, personnel, equipment, lighting schedules, etc. In this paper, EnergyPlus is used for load calculation based on the ideal air-conditioning system settings of the corresponding building energy consumption model, and the results of the building cooling and heating loads are obtained. Figure 11 shows the calculated values of cooling load indexes for each functional partition of the air-conditioning area of the processed building plane.



Case 1 Case 2 Case 3

Figure 11. Results of cooling load (W/m^2) for each air conditioning zone of the buildings' plane.

Based on the design value of the cold load index of each functional partition of the case building, the HVAC system cooling and heating source and terminal equipment selection calculation can be carried out for the case building, taking Case 1 as an example. In this paper, we selected the Boruvka minimum spanning tree algorithm and the minimum Steno tree algorithm and generated four weighted diagrams based on four different boundary conditions as input conditions for the minimum spanning tree algorithm, as well as the minimum Steiner tree, as follows:

1. Completely equally weighted graphs;
2. Restriction of the node out-degree of the equipment room to one, i.e., connecting lines outward in only one direction at that point;
3. Non-equal-weighted graph, setting the weight value of air-conditioned areas higher than that of the non-air-conditioned areas. (4) The same as in (3);
4. The same non-equal-weighted graph as in (3), while restricting the node out-degree of the equipment room to one.

For the three case building models selected in this section, the results of the HVAC terminal water system equipment layout divided into eight sample classes corresponding to each case building plan are shown in Figure 12, respectively. As can be seen in the figure, eight different terminal piping layout results can be generated for each of the three cases using the automated design methodology of this study. The results generated based

on Boundary Conditions 1 and 2 and Boundary Conditions 3 and 4 have only localized layout differences at the equipment room nodes. The results show that Sample Classes 3 and 4 utilize more of the corridor area in the building floor plan for layout, which is a better design rationalization than that of Sample Classes 1 and 2.

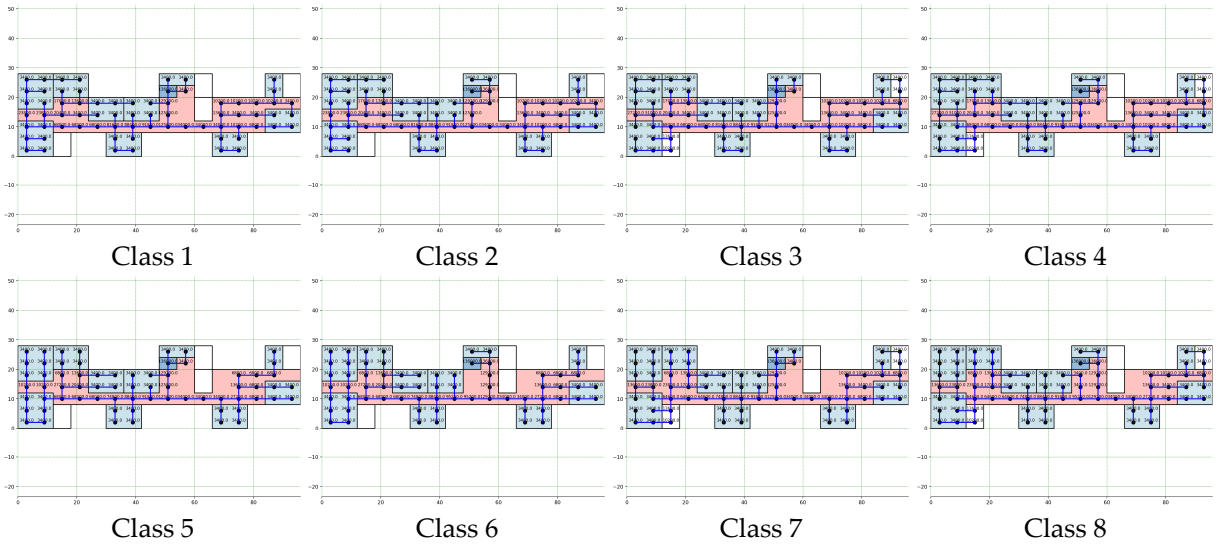


Figure 12. Schematic diagram of system terminal piping layout for building Case 1.

5.3. System Solution Generation Results

In this paper, by using the deep graph generative model for the HVAC water system terminal equipment topology connection scheme generation method, the confidence coefficient of the model is optimized. The confidence coefficient is defined in mathematics as the degree of confidence that a sample represents the whole. Similarly, the confidence coefficient in this study represents the degree of confidence in the probability of occurrence of edges between the nodes in a graph. For example, when the confidence coefficient takes the value of 0.5, the elements in the adjacency matrix of the graph with the probability values greater than 0.5 will be considered as edges. The confidence value of the GAN model in the scheme generation results is set to 0.7, according to the rule that the out-degree of a node in the graph is at least 1. The model prediction results of the HVAC terminal water system layout of the test cases are shown in Figure 13. In addition to the GAN model, five other models were also tested in this study using the same methodology.

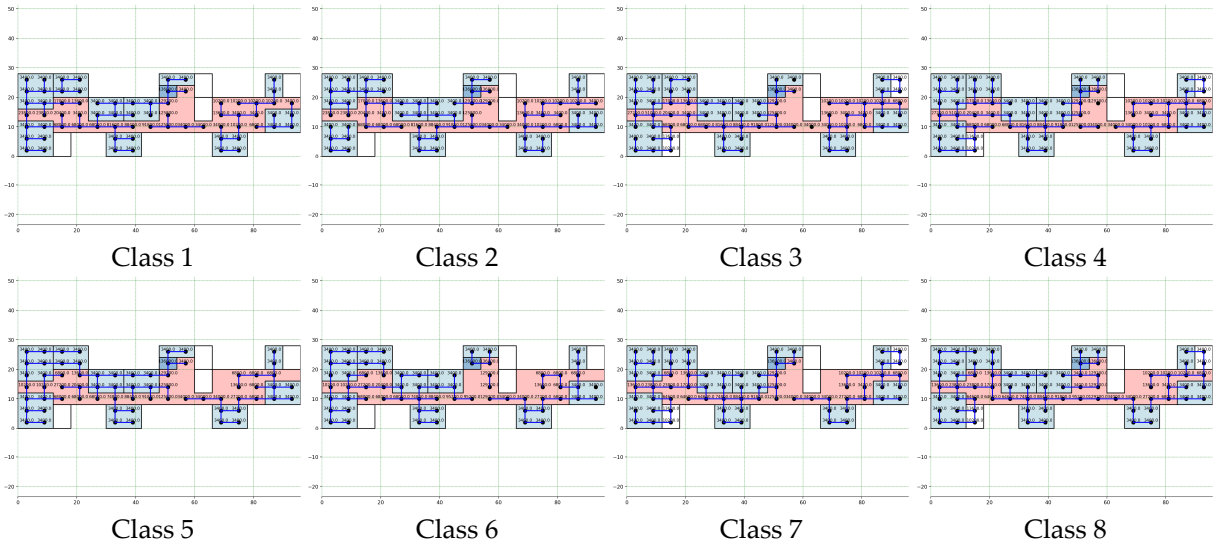


Figure 13. The GAN model predicts the final water piping layout of building Case 1.

In this study, the model prediction results of the deep graph generative model are evaluated, and the reference for the evaluation of the scenario generation results is the HVAC water system terminal equipment topology connection scenario generated based on the boundary conditions corresponding to Sample Class 4 using the automated design method shown in Figure 12. The evaluation metrics are selected to evaluate the accuracy of the graph structural data adjacency matrix of the predicted scenario with respect to the adjacency matrix of the deep graph generative model prediction results in accuracy and precision, as shown in Tables 3 and 4, and the table information visualization of statistical graphs, as shown in Figures 14 and 15.

Table 3. Accuracy of prediction results of deep graph generative models (%).

	Model	1	2	3	4	5	6	7	8	Average	Case Average
Case 1	GAN	98.3	98.4	99	98.9	97.9	97.9	98.9	98.6	98.5	98.5
	GAE	98.1	98.4	98.9	98.9	98.3	98.2	98.9	99	98.6	
	VGAE	98.3	98.3	98.9	98.8	98.2	98.3	98.6	98.6	98.5	
	GNF	98.7	98.4	98.8	99.1	98	98.1	98.7	98.8	98.6	
	VGNF	98	98.5	98.8	99	97.9	97.9	98.6	98.7	98.4	
	Diffusion	98.6	97.3	99	98.2	97.6	98.4	98.1	99.1	98.3	
Case 2	GAN	98.8	98.8	99.9	99.2	97.8	97.8	99	99	98.8	98.9
	GAE	98.8	98.6	99.8	99.5	98.4	98.6	99.2	99.1	99.0	
	VGAE	99.5	98.6	99.3	99.5	98.4	98.5	99.2	99.4	99.1	
	GNF	100	97.5	100	97.5	98.6	98.6	99.2	99.1	98.8	
	VGNF	98.8	99.3	99.6	99.6	97.8	98.3	99.1	98.9	98.9	
	Diffusion	98.6	99.2	99.5	99.1	97.6	98.5	99.6	98.2	98.8	
Case 3	GAN	98.4	98.8	99.4	99.4	98.1	98.1	99.4	99.5	98.9	99.3
	GAE	99.2	98.2	99.5	99.7	99.5	99.5	99.4	99.4	99.3	
	VGAE	99.2	99.1	99.4	99.4	99.6	99.5	99.3	99.5	99.4	
	GNF	99.3	99.7	99.6	99.7	99.4	99.7	99.7	99.7	99.6	
	VGNF	98.9	99.1	99.5	99.6	99.7	99.5	99.7	99.5	99.4	
	Diffusion	98.2	99.5	99.2	99.4	99.5	99.7	99.7	99.5	99.3	

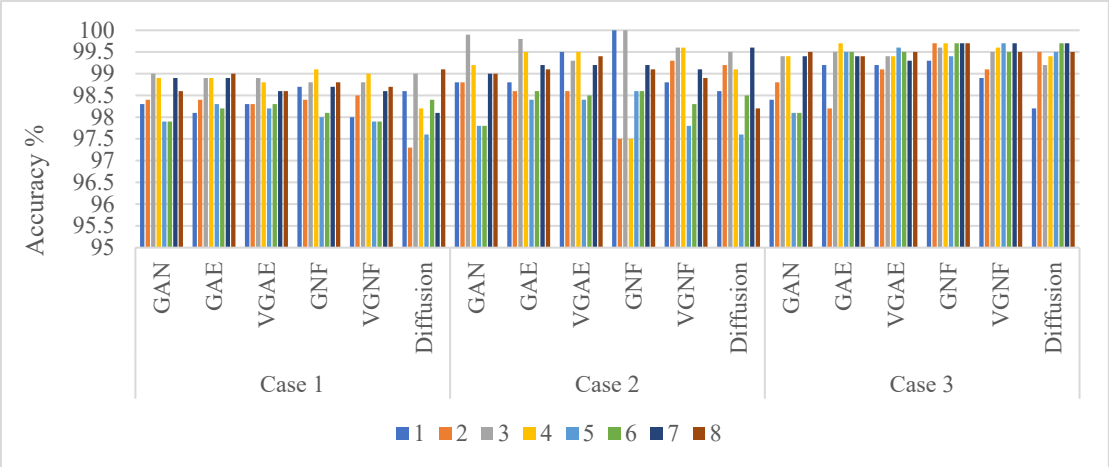


Figure 14. Accuracy statistics of the prediction results of the deep graph generative model.

The results presented in Table 3 show that, when using the deep graph generative models of this study with building spatial topology information as the input, the prediction result accuracy of all models averaged 98.5%, 98.9%, and 99.3% in the three cases,

respectively. For each test case, the average accuracy of the predictions using different models on the eight sample class boundary conditions is also above 98.3%, 98.8%, and 98.9%, respectively. As can be seen from the distribution of the model prediction result accuracies shown in Figure 14, the models used in this study generally outperformed the other two cases in terms of prediction result accuracies for Case 3. From the overall distribution of data on the graph, in the case of positive design solutions generated based on eight different boundary conditions against each other, the six deep graph generative models used in this study all achieved more than 97% in terms of prediction accuracy, which indicates that the prediction results of the deep graph generative models used in this study have an ideal accuracy.

Table 4. Precision of prediction results of deep graph generative models (%).

	Model	1	2	3	4	5	6	7	8	Average	Case Average
Case 1	GAN	65.8	67.1	79.7	77.2	56.9	58.2	77.2	70.9	69.1	69.8
	GAE	62	67.1	77.2	78.5	65.8	63.3	77.2	79.7	71.4	
	VGAE	64.6	64.6	77.2	75.9	63.3	65.8	72.2	72.2	69.5	
	GNF	72.2	68.4	75.9	82.3	59.5	60.8	73.4	74.7	70.9	
	VGNF	59.5	69.6	76	79.7	58.2	57	72.2	73.4	68.2	
	Diffusion	66.5	62.4	75.4	80.5	60.3	62.5	72.2	76.5	69.5	
Case 2	GAN	75.3	75.3	97.4	84.4	57.1	57.1	80.5	80.5	76.0	80.0
	GAE	75.3	72.7	96.1	90.9	67.5	71.4	84.4	83.1	80.2	
	VGAE	89.6	71.4	87	89.6	67.5	70.1	84.4	88.3	81.0	
	GNF	100	97.5	100	94.6	71.4	71.4	84.4	81.8	87.6	
	VGNF	75.3	85.7	92.2	92.2	57.1	66.2	81.8	79.2	78.7	
	Diffusion	65.3	75.5	82.6	86.2	65.1	73.2	79.8	84.2	76.5	
Case 3	GAN	68.4	76.3	88.2	88.2	63.2	63.2	88.2	89.5	78.2	86.7
	GAE	84.2	64.5	90.8	93.4	90.8	89.5	88.2	88.2	86.2	
	VGAE	84.2	81.6	88.2	88.2	92.1	90.8	86.8	89.5	87.7	
	GNF	86.8	93.4	92.1	93.4	88.2	94.7	94.7	93.4	92.1	
	VGNF	78.9	82.9	89.5	92.1	93.4	89.5	93.4	90.8	88.8	
	Diffusion	75.3	85.4	87.3	84.6	90.5	86.8	94.6	91.6	87.0	

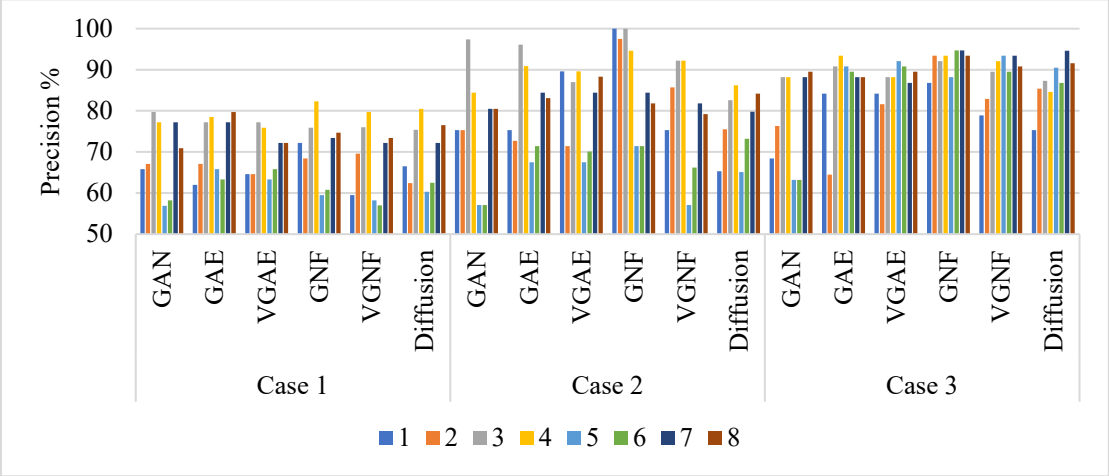


Figure 15. Precision of prediction results of deep graph generative models (%).

The results presented in Table 4 show that the accuracy of the prediction results of the deep graph generative model on the different case buildings varies considerably, with the average accuracy for the three cases being 69.8%, 80.0%, and 86.7%, respectively. For each test case, the average accuracy of the predictions of the eight sample class boundary

conditions using different models was above 68.2%, 76.0%, and 78.2%, respectively. The accuracy of model prediction is used to characterize how accurately the model predicts the positive sample results, and it can be seen that the deep graph generative model has some differences in predicting the layout details of the design results with the positive design results of its control, which are manifested in the missing and ectopic piping of the local piping layout. As can be seen from the distribution of the accuracy of the model prediction results presented in Figure 15, the accuracy of the model used in this study in predicting the results in Case 3 is generally better than that of the other two cases. From the overall distribution of data on the graph, in the case of positive design solutions generated based on eight different boundary conditions against each other, most of the six deep graph generative models used in this study exceeded 60% in prediction accuracy; moreover, only a small number of the models performed poorly in prediction results, which indicates that the prediction results of the deep graph generative models used in this study have good accuracy.

The three case buildings used for testing in this section all had existing HVAC water system design results that had been manually designed by the designers. The HVAC water system design results of the case buildings were simplified and integrated into the graph structure data presented in this study to facilitate their comparison with the automated design and model prediction results.

5.4. Experimental Results

In this study, the automated design results generated using eight different boundary conditions are compared with the transformation results of the actual design schemes of the test case buildings to evaluate the automated design generated schemes. The evaluation metrics are selected as the accuracy and precision metrics of the adjacency matrix of the graph structure data of the automated design scheme relative to the adjacency matrix of the actual design scheme of the building. The calculation can obtain the accuracy and precision of the results of the automated design scheme generated using different boundary conditions, as shown in Tables 5 and 6.

From Table 5, it can be seen that the accuracy of the automated design schemes compared to the actual design schemes all exhibit high levels of accuracy, being 98.5%, 98.6%, and 99.1% for the three cases, respectively, and the accuracy performance on the different sample classes is above 90%. In contrast, the accuracy of the automated design solutions compared to the actual design solutions shown in Table 6 is relatively low, and the range of accuracy is different for different cases, with the accuracy range of Case 1 being 50–80%, Case 2 being 60–80%, and Case 3 being 60–95%. Case 1 is characterized by its relatively complex building space layout. The results of the HVAC system terminal piping layout are relatively complex. From Figures 10 and 12, a comparison can be seen, where the actual design of the terminal piping layout scheme is simpler, and there is no piping through the stairwell and the other non-air-conditioning areas, so the accuracy of its normal design results is lower. Case 2 is characterized by the existence of a larger atrium area as a compound connectivity area in its building plan space layout. By comparing Figures 10 and 12, it can be seen that the automated design method is applicable to the existence of larger compound connectivity domains; however, with the increase of the relative distance between the position of the terminals and the pipe wells, the weight of the air-conditioning area in the graph data decreases according to its proportion of the whole plan layout. And it is easy to see more longer piping layouts are located in the air-conditioning areas. Case 3 is characterized by a larger building plan layout, a larger number of air-conditioning rooms in the plan, and a longer stroke of the trunk pipe in the terminal piping layout of the HVAC system, so there is also a long piping layout in the air-conditioning area, but the overall trend of the layout is better than that found in Case 2, so the accuracy index of the program is relatively high.

Table 5. Accuracy of the automated design solution compared to that of the actual design solution (%).

Case	1	2	3	4	5	6	7	8	Average
Case 1	98.1	98.2	98	98.1	98.8	99.1	98.7	98.9	98.5
Case 2	98.2	98.3	98.4	98.4	98.6	98.7	99.1	99.1	98.6
Case 3	99.3	99.5	99.4	99.5	98.1	98.2	99.5	99.6	99.1

Table 6. Precision of the automated design solution compared to that of the actual design solution (%).

Case	1	2	3	4	5	6	7	8	Average
Case 1	54.4	57	53.2	55.7	70.9	75.9	67.1	70.9	63.1
Case 2	61.4	62.3	63.6	64.9	68.8	70.1	77.9	79.2	68.5
Case 3	86.9	89.5	86.8	89.5	61.8	64.5	89.5	92.1	82.6

In summary, for the three test case buildings selected in this section, the program obtained by the automated design method based on this study has a higher accuracy compared to that of the actual design program, i.e., the design method can satisfy the global trend of program design. However, in terms of accuracy, it is affected by factors such as the complexity of the spatial layout of the building and the number of rooms and functional partitions, which perform differently in the three test case buildings, with the accuracy of the results being higher in Case 3 than in the other two test buildings.

This paper evaluates the model prediction results of the deep graph generative model, the scheme generation results are evaluated with reference to the topology simplification results of the actual HVAC water system terminal equipment topology connection scheme of the test building, and the evaluation indexes are selected as the accuracy and precision indexes of the adjacency matrix of the map structure data of the predicted scheme relative to the adjacency matrix of the automated design scheme. The calculation can be obtained using different deep graph generative model prediction results in accuracy and precision situations, as shown in Tables 7 and 8.

Table 7. Accuracy of the deep graph generative model predictions relative to that of the actual design results (%).

	Model	1	2	3	4	5	6	7	8	Average	Case Average
Case 1	GAN	97.6	97.8	98.2	98.1	97.8	97.9	98.4	98.3	98.0	98.1
	GAE	98	97.9	98.1	98.3	98.3	98.1	98	98.3	98.1	
	VGAE	98.1	98.1	97.9	98.3	98.3	98.1	97.9	98.4	98.1	
	GNF	97.8	98.1	98	98	98.3	98.2	98.1	98.4	98.1	
	VGNF	98.1	97.9	98	97.9	98.1	98.1	98.1	97.9	98.0	
	Diffusion	98.6	97.3	99	98.2	97.6	98.4	98.1	99.1	98.3	
Case 2	GAN	98.2	98.3	98.4	98.4	98.6	98.1	99.1	99.1	98.5	98.5
	GAE	97.8	98.2	98.4	98.8	97.6	97.6	98.6	98.4	98.2	
	VGAE	97.9	97.9	98.4	98.6	98.2	98.4	98.6	98.5	98.3	
	GNF	98.5	98.7	98.6	98.9	98.2	98.2	99	98.9	98.6	
	VGNF	98.1	97.8	98.5	98.4	98.2	98.7	98.7	98.4	98.4	
	Diffusion	98.6	99.2	99.5	99.1	97.6	98.5	99.6	98.2	98.8	
Case 3	GAN	98.3	98.8	99.4	99.4	97.9	97.8	99.4	99.2	98.8	99.0
	GAE	99.1	98	99.5	99.4	98.1	98	99.5	99.4	98.9	
	VGAE	99.2	99	99.5	99.4	98	98.1	99.6	99.5	99.0	
	GNF	99.2	99.5	99.6	99.6	98.1	98	99.5	99.6	99.1	
	VGNF	99.2	98.9	99.6	99.6	98.1	98	99.5	99.6	99.1	

	Diffusion	98.2	99.5	99.2	99.4	99.5	99.7	99.7	99.5	99.3

The results presented in Table 7 show that, when using the deep graph generative model in this study with the building spatial topology information as the input, the accuracy of the model's prediction results averaged 98.1%, 98.5%, and 99.0% in the three cases, respectively, which are all above 90%. For each test case, the average accuracies of the predictions using different models on the eight sample class boundary conditions were above 98.0%, 98.2%, and 98.8%, respectively. As can be seen from the distribution of the model prediction result accuracies presented in Figure 16, the models used in this study generally outperformed the other two cases in terms of the prediction result accuracies for Case 3. From the overall distribution of the data in the graph, all six deep graph generative models used in this study achieved more than 97% in prediction accuracy, and all six deep generative models used in this study performed better in prediction accuracy.

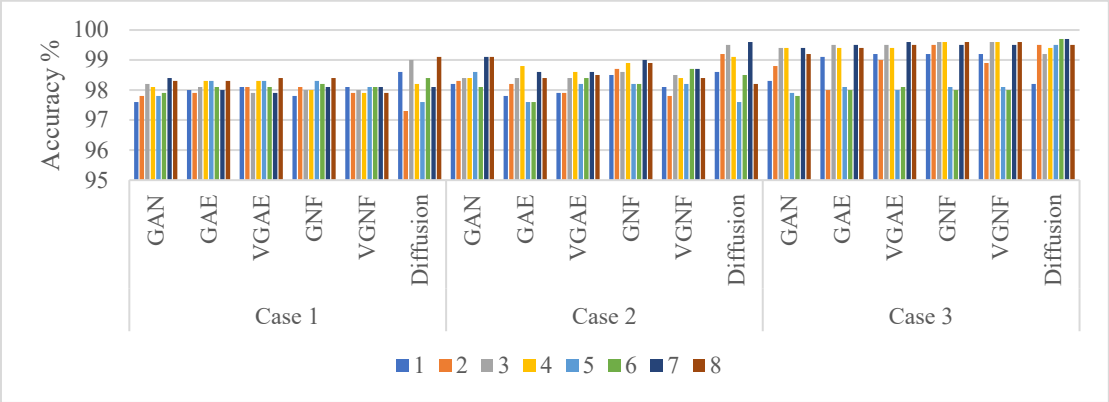


Figure 16. Deep graph generative model statistical plots of the accuracy of model predictions relative to those of the actual design results.

Table 8. Precision of the deep graph generative model predictions relative to those of the actual design results (%).

	Model	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**Model Average**	**Case Average**
Case 1	GAN	45.6	48.1	57	55.7	48.1	51.9	60.8	58.2	53.2	55.6
	GAE	53.2	51.9	55.7	58.2	58.2	54.4	53.2	59.5	55.5	
	VGAE	55.7	55.7	50.6	58.2	58.2	55.7	50.6	62	55.8	
	GNF	48.1	54.4	53.2	53.2	58.2	57	55.7	60.8	55.1	
	VGNF	54.3	50.6	53.2	51.9	55.7	54.4	54.4	51.9	53.3	
	Diffusion	56.5	52.4	65.4	60.5	60.3	62.5	62.2	66.5	60.8	
Case 2	GAN	61	62.3	63.6	64.9	68.8	70.1	77.9	79.2	68.5	64.7
	GAE	53.2	61	64.9	72.7	49.4	48.1	67.5	63.6	60.1	
	VGAE	54.5	54.5	63.6	67.5	59.7	64.9	68.8	66.2	62.5	
	GNF	66.2	70.1	68.8	74.2	59.7	61.4	75.3	74	68.7	
	VGNF	59.7	53.2	67.5	63.6	59.7	59.7	70.1	64.9	62.3	
	Diffusion	65.3	55.5	72.6	66.2	65.1	63.2	69.8	74.2	66.5	
Case 3	GAN	67.1	75	88.2	89.5	57.9	56.6	86.8	84.2	75.7	79.8
	GAE	81.6	60.5	89.5	86.8	63.2	60.5	89.5	88.2	77.5	
	VGAE	82.9	78.9	89.5	86.8	61.8	63.2	90.8	89.5	80.4	
	GNF	82.9	89.5	90.8	90.8	63.2	60.5	89.5	92.1	82.4	
	VGNF	82.9	77.6	92.1	90.1	63.2	61.8	89.5	90.8	81.0	
	Diffusion	85.3	75.4	87.3	94.6	60.5	66.8	94.6	91.6	82.0	

The results presented in Table 8 show that the accuracy of the prediction results of the deep graph generative model varies considerably across the sample types, with the average accuracy for the three cases being 55.6%, 64.7%, and 79.8%, respectively. For each test case, the average accuracy of the predictions over the eight sample class boundary conditions using the different models is above 53.2%, 60.1%, and 75.7%, respectively. From Figure 17, it can be seen that the prediction accuracies of the six models perform better in Case 3 than in the other two cases, and the distribution of the accuracies of all of the predictions shows that most of the predictions have accuracies higher than 50%.

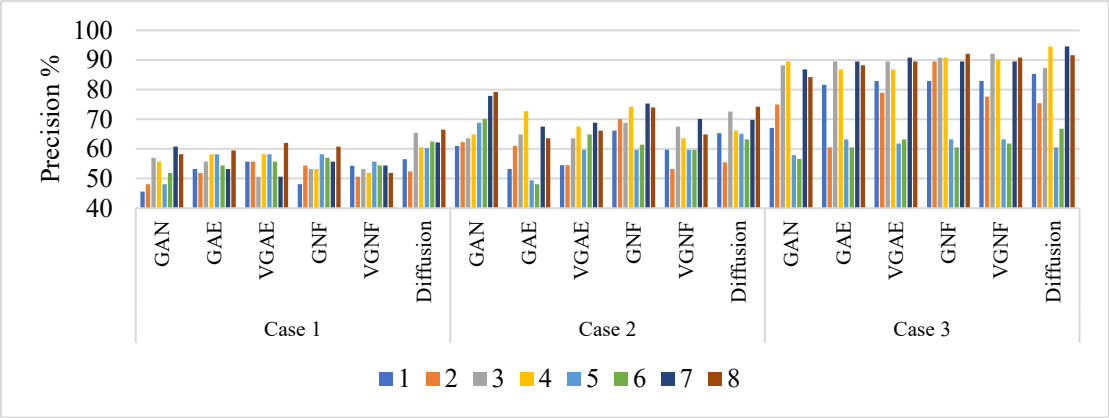


Figure 17. Deep graph generative models' statistical plots of the precision of the model's predicted results relative to those of the actual design results.

The model prediction results, automated design results, and real design results of the three cases are input into the GNN-based scheme evaluation model trained in previous research for scoring [46]. We used four subjective and objective evaluation indexes to evaluate the results of terminal piping layouts generated by the generative models, which are hydraulic balance index (Hi), the economical index (Ei), the rationality index (Ri), and the aesthetics index (Ai). The scoring results of the four indexes are shown in Table 9. Taking Case 1 as an example, the statistical graph of the scoring results of the four indicators of the terminal piping design program is shown in Figure 18.

Table 9. Scoring results for the four metrics of the case building terminal piping design program.

Type	Case 1				Case 2				Case 3			
	Hi	**Ei**	**Ai**	**Ri**	**Hi**	**Ei**	**Ai**	**Ri**	**Hi**	**Ei**	**Ai**	**Ri**
Actual design	78	75	85	92	80	76	89	95	80	85	89	97
Automated design	75	72	80	90	77	54	88	92	76	80	84	95
MLP	69	71	77	86	72	73	79	88	67	68	75	83
GNN	65	66	73	81	65	67	74	82	67	69	75	84
GraphSAGE	69	71	77	86	70	72	78	87	68	70	76	85
GAT	69	70	77	85	72	73	80	88	68	70	77	85
TransformerGNN	71	73	79	88	70	72	78	87	68	70	77	86
GCN	70	72	78	87	71	73	79	88	65	67	74	83
TAG	72	74	78	88	71	72	77	86	72	73	78	87

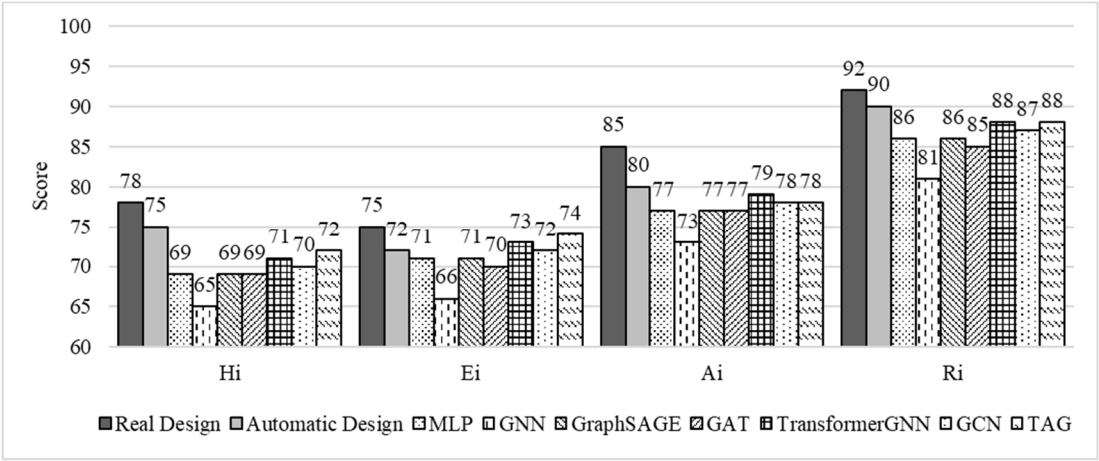


Figure 18. Statistical chart of the scoring results of the four indicators of the terminal piping design scheme in Case 1.

As shown in Table 9, the prediction results of the deep graph generative model for the three test building cases were scored using the scheme evaluation method based on the GNN model and compared with the scoring results of the real design and the positive design. It can be seen that the scoring results of the real design and the scoring results of the automated design are higher than the prediction results of the deep graph model. However, from Figure 18, it seems that the difference in scoring results between the design results generated using the deep graph generative model is small, and the difference between these scores and the scores of the real design results is within 15 points.

As shown in Table 10, the program evaluation results of Cases 1, 2, and 3 have close ratings on Sample Classes 1 and 2, and similarly also appear in Sample Classes 3 and 4, 5 and 6, and 7 and 8. It can be seen from Figure 19 that the ratings of Sample Classes 3 and 4 are higher than those of Sample Classes 1 and 2, while the ratings of Sample Classes 7 and 8 are higher than those of Sample Classes 5 and 6. Combining the analysis of relevant laws presented in Section 5.2, the terminal piping layouts generated by the deep graph generative model effectively reflect the automated positive HVAC system and potential design laws in the design scheme. At the same time, the program evaluation method based on GNNs in this study also gives program scoring results that meet the designer's requirements for the terminal piping layout of the test case building.

Table 10. Scoring performance of design solutions generated from GAN-based models on different sample classes.

Type	Case 1				Case 2				Case 3			
	Hi	Ei	Ai	Ri	Hi	Ei	Ai	Ri	Hi	Ei	Ai	Ri
1	58	64	61	70	55	58	65	74	57	59	66	74
2	59	64	62	80	56	57	65	73	57	59	68	74
3	65	66	73	75	64	67	72	82	62	66	73	84
4	65	66	73	81	65	67	74	82	67	69	75	84
5	62	64	71	79	54	58	73	74	58	58	67	74
6	62	64	71	79	57	59	73	73	56	60	65	75
7	73	74	81	91	70	72	79	92	70	73	79	92
8	74	75	82	91	75	77	83	91	71	73	79	92

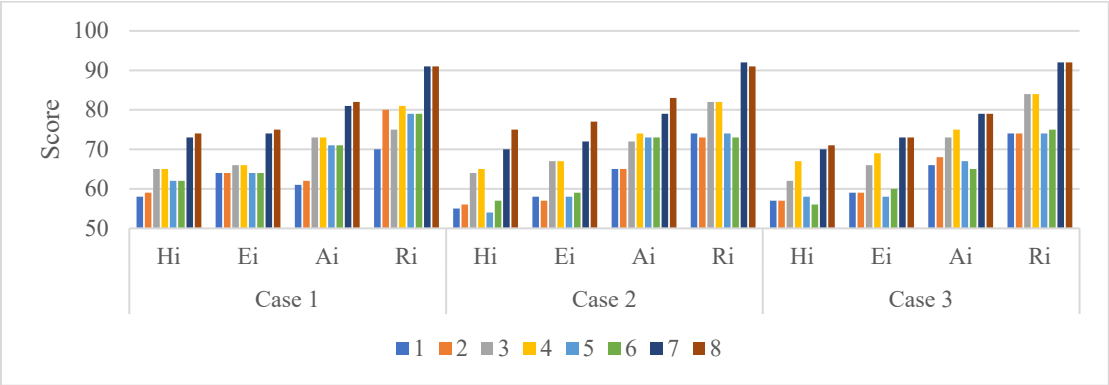


Figure 19. Statistical plot of the performance of the scores of the design solutions generated by the GAN-based model on different sample classes.

6. Discussion

This study selected three real-world building models to validate the automated design method and case-based learning approach proposed for HVAC systems. A method was developed to simplify and abstract the floor plan layouts of the actual test buildings, allowing the comparison and evaluation of the design results generated by the automated design process and those derived from the deep graph generative models. These results were stored in a unified graph data structure to facilitate a systematic assessment of the feasibility and generalizability of the proposed methods.

This study focuses on comparing the differences between the water system terminal pipeline layouts predicted by the deep graph generative models, the results from the automated design method, and the actual design outcomes. The differences were quantified using accuracy and precision metrics based on the adjacency matrix. The experimental results indicate that the solutions generated by the automated design method exhibit high accuracy when compared to the actual design plans, consistently achieving over 90% accuracy. This suggests that the automated design method is capable of capturing the overall design patterns. However, precision varied significantly across the three test buildings, influenced by factors such as the complexity of spatial layouts and the number of rooms and functional zones, with precision rates ranging from 50% to 95%. Notably, Test Case 3 demonstrated higher precision than the other two cases.

The deep graph generative models, which utilized building spatial topology as the input, also achieved over 90% accuracy across all three cases, indicating a robust performance across the six models used in this study. However, the precision of the model predictions varied considerably across the different sample types, with Test Case 3 consistently outperforming the other two cases. Particularly, the models performed better on sample types 3, 4, 7, and 8, which correspond to layouts more closely aligned with the actual building configurations. Among the various deep graph generative models, the VGAE model demonstrated superior predictive performance across multiple sample classifications in all three cases.

Furthermore, this study employed a GNN-based evaluation method to score the layout predictions of the deep graph generative models. These scores were then compared with the scores of the actual design results and those from the automated design method. The findings revealed that the scores of the actual and automated design results were higher than those of the deep graph model predictions. The score differences among the deep-graph-generated designs were minimal, with deviations from the actual design scores within 15 points. An analysis of the scoring across the different sample types showed that the evaluation results for Cases 1, 2, and 3 were similar for Sample Types 1 and 2, as well as for Types 3 and 4, 5 and 6, and 7 and 8. Notably, Sample Types 3 and 4 received higher scores than Types 1 and 2, while Types 7 and 8 outperformed Types 5 and 6.

Combining these findings with the analysis presented in Section 5, it is evident that the water system terminal piping layouts generated by the deep graph models effectively reflect the underlying design patterns captured by the automated HVAC system design method. Additionally, the GNN-based evaluation method provided satisfactory scores for the terminal pipeline layouts in the cases, aligning with the expectations of professional designers.

7. Conclusions and Further Works

This research explored the application of deep graph generative algorithms to the system-level generation of terminal water system designs in HVAC systems. Six models were trained on graph datasets, with hyperparameters being optimized through random search. The models, trained in design schemes under eight boundary conditions, achieved over 90% accuracy and 75% precision in design results prediction compared to the automated forward design results. The deep graph generative models trained on datasets from different sample classes exhibited both global and local differences in terminal piping layouts. This indicates that these models can recognize both overall design changes and localized modifications made by designers, although they are more sensitive to global layout variations than to local updates.

Three real-world building models were selected to validate the automated design method and case-based learning approach proposed in this study. An equivalent model was introduced to simplify and grid the floor plan layouts of the test buildings. The experimental results revealed that the solutions generated by the automated design method achieved higher accuracy compared to the actual design schemes, which suggest that the method effectively captures global design patterns. However, precision varied across the three test buildings due to factors such as spatial layout complexity and the number of rooms and functional zones, with precision ranging from 50% to 95%, in which Case 3 exhibited higher precision than the other two cases. The deep graph generative models, which took spatial topology information as the input, consistently achieved over 90% accuracy across the three cases, demonstrating a robust performance across all six models employed in this study. However, precision varied significantly among the different sample types, with the models showing superior performance in Case 3. Particularly, the models performed better in Sample Types 3, 4, 7, and 8, which correspond to layouts more closely aligned with the actual building configurations. Among the various deep graph generative models, the VGAE model demonstrated the best performance across multiple sample classifications in all three cases.

Additionally, the eight different terminal pipeline layout schemes generated by the deep graph models were evaluated using the GNN-based evaluation model proposed in this study. The scoring results effectively reflected the design logic and patterns across the different sample classes. Therefore, the deep graph generative models were able to learn the underlying rules and logic in the HVAC system design schemes from the graph-based dataset created in this study, and these rules were well reflected in the HVAC water system design outcomes for the test buildings.

The parametric design of HVAC systems in the future will be more intelligent, and, by combining technologies such as artificial intelligence and machine learning, parametric design systems will also be able to automatically learn and optimize design parameters to better meet design needs. At the same time, future parametric design will involve more multimodal design to adapt to the design needs of diverse HVAC systems, i.e., multiple design options will be considered in a single parametric model, thus providing more choices and flexibility.

This study is currently only applicable to office building models with simple and small-scale building plan layouts. In the future, more refined meshing methods can be considered to improve the applicability of the graph-structured data model for buildings with complex plan layouts. At the same time, the node and edge features used to describe the building information and HVAC information in the graph structure data can be

increased, so that the graph structure data can portray the building and its HVAC system more comprehensively.

Author Contributions: Conceptualization, H.W., P.X. and J.G.; methodology, H.W.; software, H.W.; validation, H.W.; formal analysis, H.W.; investigation, H.W.; resources, H.W.; data curation, H.W.; writing—original draft preparation, H.W.; writing—review and editing, H.W. and R.J.; visualization, R.J.; supervision, P.X.; project administration, P.X.; funding acquisition, P.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (No.52161135202).

Data Availability Statement: The original contributions presented in the study are included in the article, further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sacks, R.; Eastman, C.; Lee, G.; Teicholz, P. *BIM Handbook: A Guide to Building Information Modeling for Owners, Designers, Engineers, Contractors, and Facility Managers*; John Wiley & Sons: Hoboken, NJ, USA, 2018; ISBN 978-1-119-28753-7.
2. Qi, Z.; Jin, R.; Li, J.; Guan, H.; Xu, P. BIM-Based Automated Multi-Air Distribution Layout Generation for Office Buildings: A Case Study. *Buildings* **2023**, *13*, 1819. https://doi.org/10.3390/buildings13071819.
3. Sha, H.; Xu, P.; Yang, Z.; Chen, Y.; Tang, J. Overview of Computational Intelligence for Building Energy System Design. *Renew. Sustain. Energy Rev.* **2019**, *108*, 76–90. https://doi.org/10.1016/j.rser.2019.03.018.
4. Tang, X.; Zhang, J.; Liang, R. The Design of Heating, Ventilation, and Air Conditioning Systems Based on Building Information Modeling: A Review from the Perspective of Automatic and Intelligent Methods. *J. Build. Eng.* **2024**, *82*, 108200. https://doi.org/10.1016/j.jobe.2023.108200.
5. Singh, V.; Gu, N. Towards an Integrated Generative Design Framework. *Des. Stud.* **2012**, *33*, 185–207. https://doi.org/10.1016/j.destud.2011.06.001.
6. Zhang, J.; Liu, N.; Wang, S. Generative Design and Performance Optimization of Residential Buildings Based on Parametric Algorithm. *Energy Build.* **2021**, *244*, 111033. https://doi.org/10.1016/j.enbuild.2021.111033.
7. Gu, N.; Singh, V.; Merrick, K. A Framework to Integrate Generative Design Techniques for Enhancing Design Automation. In Proceedings of the 15th International Conference on Computer Aided Architectural Design Research in Asia, Hong Kong 7-10 April 2010, pp. 127-136.
8. Caetano, I.; Santos, L.; Leitão, A. Computational Design in Architecture: Defining Parametric, Generative, and Algorithmic Design. *Front. Archit. Res.* **2020**, *9*, 287–300. https://doi.org/10.1016/j.foar.2019.12.008.
9. Sönmez, N.O. A Review of the Use of Examples for Automating Architectural Design Tasks. *Comput. Aided Des.* **2018**, *96*, 13–30. https://doi.org/10.1016/j.cad.2017.10.005.
10. Stiny, G.; Mitchell, W.J. The Palladian Grammar. *Environ. Plan. B Plan. Des.* **1978**, *5*, 5–18.
11. F Downing; Flemming, U. The Bungalows of Buffalo. *Environ. Plan. B* **1981**, *8*, 269–293.
12. Müller, P.; Wonka, P.; Haegler, S.; Ulmer, A.; Van Gool, L. Procedural Modeling of Buildings. *ACM Trans. Graph.* **2006**, *25*, 614–623. https://doi.org/10.1145/1141911.1141931.
13. Shalizi, C. A New Kind of Science by Stephen Wolfram. Wolfram Media. 2002. https://www.wolframscience.com/nks/.
14. Delgarm, N.; Sajadi, B.; Delgarm, S.; Kowsary, F. A Novel Approach for the Simulation-Based Optimization of the Buildings Energy Consumption Using NSGA-II: Case Study in Iran. *Energy Build.* **2016**, *127*, 552–560. https://doi.org/10.1016/j.enbuild.2016.05.052.
15. Yu, W.; Li, B.; Jia, H.; Zhang, M.; Wang, D. Application of Multi-Objective Genetic Algorithm to Optimize Energy Efficiency and Thermal Comfort in Building Design. *Energy Build.* **2015**, *88*, 135–143. https://doi.org/10.1016/j.enbuild.2014.11.063.
16. Carlucci, S.; Cattarin, G.; Causone, F.; Pagliano, L. Multi-Objective Optimization of a Nearly Zero-Energy Building Based on Thermal and Visual Discomfort Minimization Using a Non-Dominated Sorting Genetic Algorithm (NSGA-II). *Energy Build.* **2015**, *104*, 378–394. https://doi.org/10.1016/j.enbuild.2015.06.064.
17. Hamdy, M.; Nguyen, A.-T.; Hensen, J.L.M. A Performance Comparison of Multi-Objective Optimization Algorithms for Solving Nearly-Zero-Energy-Building Design Problems. *Energy Build.* **2016**, *121*, 57–71. https://doi.org/10.1016/j.enbuild.2016.03.035.
18. Murray, S.N.; Walsh, B.P.; Kelliher, D.; O'Sullivan, D.T.J. Multi-Variable Optimization of Thermal Energy Efficiency Retrofitting of Buildings Using Static Modelling and Genetic Algorithms—A Case Study. *Build. Environ.* **2014**, *75*, 98–107. https://doi.org/10.1016/j.buildenv.2014.01.011.

19. Yu, Z.; Chen, J.; Sun, Y.; Zhang, G. A GA-Based System Sizing Method for Net-Zero Energy Buildings Considering Multi-Criteria Performance Requirements under Parameter Uncertainties. *Energy Build.* **2016**, *129*, 524–534. https://doi.org/10.1016/j.enbuild.2016.08.032.
20. Cho, J.; Shin, S.; Kim, J.; Hong, H. Development of an Energy Evaluation Methodology to Make Multiple Predictions of the HVAC&R System Energy Demand for Office Buildings. *Energy Build.* **2014**, *80*, 169–183. https://doi.org/10.1016/j.enbuild.2014.04.046.
21. Tuhus-Dubrow, D.; Krarti, M. Genetic-Algorithm Based Approach to Optimize Building Envelope Design for Residential Buildings. *Build. Environ.* **2010**, *45*, 1574–1581. https://doi.org/10.1016/j.buildenv.2010.01.005.
22. Palonen, M.; Hasan, A.; Sirén, K. A Genetic Algorithm for Optimization of Building Envelope Andhvac System Parameters. In Proceedings of the Building Simulation 2009: 11th Conference of IBPSA, Glasgow, Scotland, 27–30 July 2009.
23. Wright, J.; Farmani, R. The Simultaneous Optimization of Building Fabric Construction, HVAC System Size, and the Plant Control Strategy. In Proceedings of the 7th International IBPSA Conference, Rio de Janeiro, Brazil, 13–15 August 2001.
24. Asiedu, Y.; Besant, R.; Gu, P. HVAC Duct System Design Using Genetic Algorithms. *HVACR Res.* **2000**, *6*, 149–173. https://doi.org/10.1080/10789669.2000.10391255.
25. Berquist, J.; Tessier, A.; Brien, W.; Attar, R.; Khan, A. An Investigation of Generative Design for Heating, Ventilation, and Air-Conditioning. In Proceedings of the SIMAUD '17: Proceedings of the Symposium on Simulation for Architecture and Urban Design, Toronto, ON, Canada, 22–24 May 2017.
26. Oh, S.; Jung, Y.; Kim, S.; Lee, I.; Kang, N. Deep Generative Design: Integration of Topology Optimization and Generative Models. *J. Mech. Des.* **2019**, *141*, 1.
27. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph Neural Networks: A Review of Methods and Applications. *AI Open* **2020**, *1*, 57–81. https://doi.org/10.1016/j.aiopen.2021.01.001.
28. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Yu, P.S. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2021**, *32*, 4–24. https://doi.org/10.1109/TNNLS.2020.2978386.
29. Fan, C.; Lin, Y.; Piscitelli, M.S.; Chiosa, R.; Wang, H.; Capozzoli, A.; Ma, Y. Leveraging Graph Convolutional Networks for Semi-Supervised Fault Diagnosis of HVAC Systems in Data-Scarce Contexts. *Build. Simul.* **2023**, *16*, 1499–1517. https://doi.org/10.1007/s12273-023-1041-1.
30. Chen, Z.; Deng, Q.; Ren, H.; Zhao, Z.; Peng, T.; Yang, C.; Gui, W. A New Energy Consumption Prediction Method for Chillers Based on GraphSAGE by Combining Empirical Knowledge and Operating Data. *Appl. Energy* **2022**, *310*, 118410. https://doi.org/10.1016/j.apenergy.2021.118410.
31. Lu, J.; Zhang, C.; Li, J.; Zhao, Y.; Qiu, W.; Li, T.; Zhou, K.; He, J. Graph Convolutional Networks-Based Method for Estimating Design Loads of Complex Buildings in the Preliminary Design Stage. *Appl. Energy* **2022**, *322*, 119478. https://doi.org/10.1016/j.apenergy.2022.119478.
32. Hu, Y.; Cheng, X.; Wang, S.; Chen, J.; Zhao, T.; Dai, E. Times Series Forecasting for Urban Building Energy Consumption Based on Graph Convolutional Network. *Appl. Energy* **2022**, *307*, 118231. https://doi.org/10.1016/j.apenergy.2021.118231.
33. Gnecco, V.M.; Vittori, F.; Pisello, A.L. Digital Twins for Decoding Human-Building Interaction in Multi-Domain Test-Rooms for Environmental Comfort and Energy Saving via Graph Representation. *Energy Build.* **2023**, *279*, 112652. https://doi.org/10.1016/j.enbuild.2022.112652.
34. Xie, Y.; Stravoravdis, S. Generating Occupancy Profiles for Building Simulations Using a Hybrid GNN and LSTM Framework. *Energies* **2023**, *16*, 4638. https://doi.org/10.3390/en16124638.
35. Zhang, J.; Xiao, F.; Li, A.; Ma, T.; Xu, K.; Zhang, H.; Yan, R.; Fang, X.; Li, Y.; Wang, D. Graph Neural Network-Based Spatio-Temporal Indoor Environment Prediction and Optimal Control for Central Air-Conditioning Systems. *Build. Environ.* **2023**, *242*, 110600. https://doi.org/10.1016/j.buildenv.2023.110600.
36. Ahmedt-Aristizabal, D.; Armin, M.A.; Denman, S.; Fookes, C.; Petersson, L. Graph-Based Deep Learning for Medical Diagnosis and Analysis: Past, Present and Future. *Sensors* **2021**, *21*, 4758.
37. Xu, J.; Li, H.; Zhou, S. An Overview of Deep Generative Models. *IETE Tech. Rev.* **2015**, *32*, 131–139. https://doi.org/10.1080/02564602.2014.987328.
38. Li, Y.; Vinyals, O.; Dyer, C.; Pascanu, R.; Battaglia, P. Learning Deep Generative Models of Graphs. *arXiv* **2018**, arXiv:1803.03324.
39. Faez, F.; Ommi, Y.; Baghshah, M.S.; Rabiee, H.R. Deep Graph Generators: A Survey. *IEEE Access* **2021**, *9*, 106675–106702.
40. Kipf, T.N.; Welling, M. Variational Graph Auto-Encoders. *arXiv* **2016**, arXiv:1611.07308.
41. Maziarka, Ł.; Pocha, A.; Kaczmarczyk, J.; Rataj, K.; Danel, T.; Warchoł, M. Mol-CycleGAN: A Generative Model for Molecular Optimization. *J. Cheminform.* **2020**, *12*, 2. https://doi.org/10.1186/s13321-019-0404-1.
42. Kobyzev, I.; Prince, S.J.D.; Brubaker, M.A. Normalizing Flows: An Introduction and Review of Current Methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 3964–3979. https://doi.org/10.1109/TPAMI.2020.2992934.
43. Yang, L.; Zhang, Z.; Song, Y.; Hong, S.; Xu, R.; Zhao, Y.; Zhang, W.; Cui, B.; Yang, M.-H. Diffusion Models: A Comprehensive Survey of Methods and Applications. *ACM Comput. Surv.* **2023**, *56*, 1–39.
44. Lin, B.; Jabi, W.; Corcoran, P.; Lannon, S. The Application of Deep Generative Models in Urban Form Generation Based on Topology: A Review. *Archit. Sci. Rev.* **2024**, *67*, 189–204. https://doi.org/10.1080/00038628.2023.2209550.

45. Wang, H.; Xu, P.; Sha, H.; Gu, J.; Xiao, T.; Yang, Y.; Zhang, D. BIM-Based Automated Design for HVAC System of Office Buildings—An Experimental Study. *Build. Simul.* **2022**, *15*, 1177–1192. https://doi.org/10.1007/s12273-021-0883-7.
46. Wang, H.; Xu, P.; Zhang, Y.; Jin, R.; Gu, J. GNNs Based Method for Evaluating HVAC Terminal Piping Layout Solutions in Office Buildings. In Proceedings of the Building Simulation 2023: 18th Conference of IBPSA, Shanghai, China, 4 September 2023.