designs

MDPI

Article

HCVEA: Personalized Residential Layout Generation via an Improved Conditional Variational Autoencoder with Reinforcement Learning

Hongting He [1], Zunyue Liu [1,*] and Fei Xiao [2,*]

[1] College of Furnishings and Industrial Design, Nanjing Forestry University, Nanjing 210037, China; 2411403106@njfu.edu.cn
[2] Department of Civil and Airport Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China
* Correspondence: liuzunyue@njfu.edu.cn (Z.L.); fei.xiao@nuaa.edu.cn (F.X.)

Abstract

With the growing demand for personalized design, residential layout generation has become a key research area in architecture and artificial intelligence. This paper presents a novel method for generating personalized layouts using an improved Conditional Variational Autoencoder (HCVEA). This method introduces conditional variables to control key features such as room count, functional zoning, and spatial distribution, while enhancing the latent space structure to improve both diversity and controllability. By incorporating user preferences as conditional inputs and integrating reinforcement learning with the autoencoder-based architecture, the layout generation process is optimized for more accurate and efficient results. A novel Connectionist Temporal Classification Attention (CTC-Attention) decoder is introduced to improve contextual semantic understanding. The Asynchronous Advantage Actor–Critic (A3C) reinforcement learning algorithm is employed to enhance training efficiency. Experimental results averaged over three independent runs show that HCVEA achieves an FZMR of 90.7% and an RAR of 92.3% with low standard deviation, outperforming baseline models. It also maintains a constraint compliance rate of 88.6% and adaptability to different room configurations.

Keywords: personalized design optimization; Conditional Variational Autoencoder; latent space modeling; generative models; reinforcement learning optimization

1. Introduction

In recent years, the demand for personalized design has grown rapidly, driven by increasingly diverse user preferences, lifestyle patterns, and functional requirements, especially in the fields of architecture and urban planning [1,2]. Residential layout design is no longer expected to only satisfy basic spatial constraints but also to support personalized living habits, functional rationality, and adaptive use of space [3,4]. This practical demand has highlighted the limitations of traditional one-size-fits-all design paradigms and motivated the exploration of intelligent, data-driven layout generation methods capable of balancing personalization, efficiency, and design quality [5].

Recent advances in machine learning, particularly generative models, have shown great promise in addressing these challenges [6]. Among these, Variational Autoencoders (VAEs) have gained considerable attention for their ability to model complex data distributions and generate diverse outputs [7]. However, traditional VAEs suffer from limitations in

Check for updates

Academic Editor: Aziz Ahmed

Received: 22 January 2026
Revised: 21 February 2026
Accepted: 26 February 2026
Published: 2 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

terms of controllability, particularly when generating layouts tailored to specific user needs. To improve control over generated outputs, Conditional Variational Autoencoders [3] (CVAEs) have been introduced, which condition the model on specific variables, such as user preferences or functional requirements [8]. Despite their success in various domains, CVAEs still face challenges such as latent space redundancy and difficulty in capturing complex relationships between layout components, which are essential for generating functional and diverse designs.

Researchers have made significant progress in addressing challenges in layout generation. For example, Ye et al. [9] developed a method for generating structurally coherent house layouts with global consistency by encoding rooms and their spatial relationships in a graph structure, capturing inter-room interactions to improve layout optimization. Similarly, Tang et al. [10] proposed Graph Transformer GANs (GTGANs), which leverage graph convolutions and self-attention within a Transformer-based generator. By employing connected and non-connected node attention, GTGAN better captures both global and local relationships, resulting in optimized architectural layouts. Azmeraw et al. [11] reviewed and categorized computational methods for house floor plan generation, highlighting both procedural and deep generative approaches.

Further advancements include LGGAN, introduced by Kangjie Li et al. [12], a generative adversarial network for mechanical layout generation featuring hierarchical generation, noise reduction, and size refinement, with an active learning strategy for efficient convergence with limited data. Cheng et al. [13] developed a method for land-use layout design in virtual worlds, ensuring high-quality output through geometric and planning objectives. Chen et al. [14] proposed EcGAN, an element-conditioned GAN that enhances bounding box modeling for graphic layouts using an element mask, element condition loss, and two-step discriminators. Zhao et al. [15] introduced Layout2Im, a method that generates images from layouts by disentangling object representations into specified categories and uncertain appearances, combining word embeddings and low-dimensional vectors with convolutional LSTM. In addition, Ayçam et al. [16] investigated traditional Diyarbakir house forms and settlement patterns, demonstrating that their adaptation can improve energy efficiency and sustainability in modern housing. Finally, Zakraoui et al. [17] proposed a three-stage story-to-image generation method with object layout guidance, showing improved semantic accuracy and realism compared to existing approaches.

Although existing methods have advanced layout generation, they often optimize only one or two aspects, such as structural coherence or visual quality, neglecting holistic balance. Current models rarely address four core challenges simultaneously: (1) controllability, guided by user preferences and constraints; (2) diversity, avoiding mode collapse and supporting varied outcomes; (3) functional rationality, ensuring layouts follow realistic spatial logic; and (4) training efficiency, critical for practical deployment but often limited by complex models or slow convergence. This gap motivates a unified framework. Practical integration of generative tools into architecture and AI domains remains limited. Methods that incorporate architectural constraints, functional requirements, and user preferences provide preliminary real-world validation.

To address these gaps, this paper proposes HCVEA, a hybrid controllable variational framework for residential layout generation. It integrates an improved CVAE, a CTC-Attention decoder, and the A3C reinforcement learning algorithm. The CVAE enhances latent space expressiveness for controllability and diversity, the CTC-Attention captures complex contextual dependencies for functional rationality, and A3C improves training efficiency through parallelized optimization. By coordinating these components within a unified architecture, HCVEA simultaneously addresses the four core challenges while align-

ing with practical architectural requirements, validated through controlled experiments. Contributions of this paper include the following:

(i) Enhanced latent space structure: HCVEA introduces a flexible, structured latent space with gradient stability and conditioned prior, distinguishing it from traditional CVAEs. It reduces redundancy, prevents homogenization, and enables controllable and diverse layouts, addressing CVAE limitations in personalized layout generation.
(ii) Reinforcement learning integration: A3C encodes architectural rules in the reward, unlike prior CVAE/GAN/graph-based models. It ensures rule compliance, improves functional rationality, and aligns layouts with user preferences, targeting the limitations of rule-based and graph-based methods.
(iii) CTC-Attention decoder: We extend CTC from temporal to spatial sequences, converting 2D layout relationships into topology-aware 1D sequences and combining CTC soft alignment with attention. This addresses graph-based model limitations, enhancing functional zoning and room adjacency coherence.
(iv) A3C reinforcement learning algorithm: A parallel A3C framework is tightly coupled with CTC-Attention decoding. The reward function aligns with spatial sequence features, and the action space supports fine-grained layout adjustments, enabling closed-loop optimization that accelerates convergence and ensures architectural rule compliance.
(v) Practical integration and proof-of-concept validation: HCVEA aligns with real-world architectural needs by incorporating spatial rules, user preferences, and functional constraints. Preliminary experiments confirm its effectiveness and feasibility for practical residential layout design, overcoming the limitations of existing tools.

2. Related Work

2.1. Traditional Methods for Residential Layout Generation

Traditional methods for residential layout generation have long relied on rule-based systems and predefined templates. These approaches follow standard design principles and building codes, ensuring that room placement is functional and meets basic requirements [18]. Grid-based and modular techniques are commonly employed, where rooms are arranged in a predefined grid or using standard modules. While effective for simple designs, these methods often lack flexibility and struggle to accommodate diverse or complex user requirements.

Over the past decade, research has shifted toward computational and AI-driven layout generation. Methods using optimization, graph representations, and generative models like GANs and VAEs improve adaptability, diversity, and efficiency, while incorporating functional requirements, spatial relationships, and user preferences [19,20], moving beyond rigid rule-based constraints.

Despite these advancements, traditional methods still exhibit several core limitations. Specifically, (i) over-reliance on manually predefined architectural rules limits flexibility and prevents personalized layouts. (ii) Limited coverage hinders handling complex spatial distributions across scenarios such as residential, office, or commercial layouts. (iii) The high cost of manual adjustments leads to low generation efficiency. To address these challenges, HCVEA uses a data-driven CVAE to embed personalized constraints and integrates architectural rules into the reward function of the A3C algorithm. This approach avoids the homogenization typical of rule-based methods while ensuring that generated layouts remain functionally and architecturally rational, bridging traditional and AI-driven design.

2.2. Generative Models for Residential Layout Generation

Generative models have emerged as a promising approach for residential layout generation, leveraging data-driven techniques to learn patterns from existing designs [21]. Unlike traditional rule-based methods, these models can create diverse and customized layouts by capturing complex relationships between room sizes, functionalities, and spatial arrangements [22].

Conceptually, this research area forms a pipeline with three interconnected components. (i) Input representation covers building constraints, user preferences, and functional requirements. (ii) Generative modeling uses VAEs, GANs, or graph-based networks to learn latent layout distributions and generate new designs. (iii) Evaluation and refinement ensure layouts meet spatial coherence, functional rationality, and architectural constraints. This model illustrates how generative techniques interact with architectural knowledge and user requirements to produce feasible residential layouts.

While graph-based and Transformer models capture complex dependencies, they often struggle with continuous spatial coherence and functional zoning in dense layouts [23]. The CTC-Attention decoder overcomes these limitations by transforming 2D spatial relationships into topology-aware 1D sequences and combining soft alignment with attention, ensuring semantically coherent functional zones and accurate room adjacency. VAEs and GANs complement this by providing probabilistic diversity and realistic layout structures, adaptable to user-specific constraints.

2.3. Conditional Variational Autoencoders and Applications

CVAEs extend the traditional VAE framework by incorporating conditional variables, enabling more controlled and targeted generation of layouts. In CVAEs [24], the latent space is conditioned on input variables such as user preferences, room functions, or spatial constraints, allowing flexible, personalized layout generation [25].

However, traditional CVAE-based layout generation has key limitations: (1) latent space redundancy causes homogenized layouts; (2) shallow embedding of conditional variables weakens controllability, hindering satisfaction of user requirements like functional zoning and room count; and (3) loss functions ignore spatial characteristics, reducing coherence and functional rationality. HCVEA addresses these issues with targeted improvements: latent redundancy is reduced via latent space enhancement using a gradient stability term and conditioned prior [26]; the CVAE loss is reconstructed to align with spatial properties; and user preferences are deeply embedded in encoding and decoding, enabling actively controllable layout generation.

Unlike existing VAE regularizations, such as β-VAE or implicit methods (weight decay, dropout) that mainly constrain latent distributions or parameters, HCVEA's gradient stabilization term directly controls reconstruction sensitivity to latent perturbations. This ensures smooth, spatially coherent, and functionally rational layouts, enabling diverse and user-aligned generation while overcoming limitations of prior CVAE-based methods.

3. Preliminaries

3.1. Variational Autoencoders

In our research methodology, Variational Autoencoders (VAEs) serve as the foundational generative framework for residential layout modeling. VAEs map input layouts x to a latent representation z and reconstruct x from z via an encoder–decoder structure. This allows the model to learn a probabilistic latent space capturing essential patterns in residential layouts.

The training objective is to maximize the Evidence Lower Bound (ELBO), which consists of a reconstruction term and a Kullback–Leibler (KL) divergence term that regularizes the latent space:

$$\mathcal{L}_{\text{VAE}} = \mathbb{E}_{q(z|x)}[\log p(x|z)] - D_{\text{KL}}[q(z|x)\|p(z)] \tag{1}$$

Here, $q(z|x)$ denotes the posterior distribution of the latent variable z given x, $p(x|z)$ is the likelihood of reconstructing x from z, and $D_{\text{KL}}[q(z|x)\|p(z)]$ enforces that the posterior remains close to the prior distribution.

To enable gradient-based optimization through the stochastic latent variable, we apply the reparameterization trick [27]. The latent variable z is expressed as

$$z = \mu(x) + \sigma(x) \cdot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \tag{2}$$

where $\mu(x)$ and $\sigma(x)$ are the encoder-predicted mean and standard deviation of z, and ϵ is noise sampled from a standard normal distribution. This formulation allows sampling from the posterior while maintaining differentiability for backpropagation. In our methodology, this probabilistic latent space forms the basis for the CVAE used in HCVEA, enabling controllable and diverse residential layout generation.

3.2. Advances and Challenges in Conditional Variational Autoencoders

CVAEs extend the VAE framework by conditioning the model on additional information, such as user preferences or layout constraints. In CVAEs, the latent variable z is conditioned on both the input data x and an additional conditional variable c, which represents auxiliary information (e.g., room type or design style). The conditional posterior distribution is expressed as

$$q(z|x,c) = \frac{p(x,z|c)}{p(x|c)} \tag{3}$$

where $q(z|x,c)$ represents the posterior distribution of the latent variable z conditioned on both x and c, $p(x,z|c)$ denotes the joint distribution of x and z conditioned on c, and $p(x|c)$ is the marginal likelihood of x given c. These terms describe the relationship between the input, latent variables, and conditioning information, which is central to CVAE functionality. The objective function is

$$\mathcal{L}_{\text{CVAE}} = \mathbb{E}_{q(z|x,c)}[\log p(x|z,c)] - D_{\text{KL}}[q(z|x,c)\|p(z)] \tag{4}$$

Here, conditioning on c enables CVAEs to generate layouts aligned with specific constraints for personalized design. However, traditional CVAEs suffer from latent space redundancy, shallow conditional embeddings, and weak control over spatial and functional requirements. These issues motivate HCVEA's enhancements: latent space refinement, loss function reconstruction for spatial coherence, and deep embedding of user preferences.

3.3. Combining Autoencoders and Reinforcement Learning for Layout Generation

CVAEs offer a probabilistic generative framework but cannot fully ensure functional compliance or architectural rationality. To address this, we integrate reinforcement learning (RL), treating layout generation as a sequential decision-making task. This allows iterative refinement guided by rewards encoding architectural rules, functional zoning, and user preferences. The combined training objective is

$$\mathcal{L}_{\text{combined}} = \mathcal{L}_{\text{VAE}} + \lambda \cdot \mathbb{E}_{\pi}[\text{Reward}] \tag{5}$$

where $\mathcal{L}_{\text{VAE}}$ is the VAE loss, λ balances the VAE loss and RL reward, and $\mathbb{E}_{\pi}[\text{Reward}]$ is the expected reward under policy π. The RL agent uses policy gradient methods:

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi_{\theta}}[\nabla_{\theta} \log \pi_{\theta}(s_t) R_t] \tag{6}$$

This integration justifies our research design: the CVAE captures data distributions and latent layout patterns, while RL ensures that generated layouts adhere to practical constraints and functional requirements. Together, they provide a flexible, controllable, and optimized framework for residential layout generation, addressing the limitations of purely generative or rule-based approaches. The model iteratively explores and exploits the latent space, producing diverse, feasible, and user-aligned layouts. A summary of key variables is given in Table 1.

Table 1. Summary of key variables.

Variable	Description
$q(z\|x)$	Posterior of latent variable z given input layout x
$p(x\|z)$	Likelihood of reconstructing x from z
D_{KL}	KL divergence between $q(z\|x)$ and $p(z)$
$\mu(x)$	Mean of latent variable z from encoder
$\sigma(x)$	Standard deviation of z from encoder
ϵ	Noise vector sampled from $\mathcal{N}(0, I)$
$q(z\|x, c)$	Posterior of z conditioned on x and auxiliary input c
$p(z\|c)$	Prior of latent variable z conditioned on c
z	Latent variable representing layout features in latent space
$\alpha(z)$	Attention weights over rooms in CTC-Attention decoder
c	Conditional input representing user preferences or layout constraints
$\pi_{\theta}(s_t)$	Policy function in RL at state s_t
R_t	Reward at step t for RL agent
α_{trl}	Transformer learning rate (optimization section)
α_{loss}	Spatial loss weight (loss function)
γ_{3D}	3D gamma correction factor (NeRF rendering)
γ_{opt}	Adam optimizer decay factor
λ_{reg}	Weight decay regularization coefficient
λ_{pos}	Positional encoding wavelength (Transformer input embedding)
λ_{γ}	Gradient stabilization hyperparameter for latent space
G_t	Cumulative discounted reward in A3C: $G_t = \sum_{k=t}^{T} \gamma^{k-t} R_k$

4. Methodology

4.1. Model Framework Overview

The HCVEA model integrates advanced techniques such as CVAE, RL, and a novel decoding mechanism to generate personalized and flexible residential layouts. By combining generative modeling with optimization, the model overcomes the limitations of traditional methods, enabling highly adaptable designs that meet both functional and aesthetic requirements. The framework operates through several key stages, each contributing to the creation of diverse and contextually coherent floor plans, as illustrated in Figure 1.

- Input Conditioning and Latent Space Encoding: The model conditions on user preferences and design constraints, such as room types, spatial distributions, and functional zoning, which guide the encoder to map data into a structured latent space. The encoder uses three Conv2d layers (3→64→128→256 channels, kernel = 3, stride = 2, padding = 1) followed by two fully connected layers projecting the 256 dim convolutional features to a 128 dim latent space for mean and variance, ensuring structured and diverse latent representations.

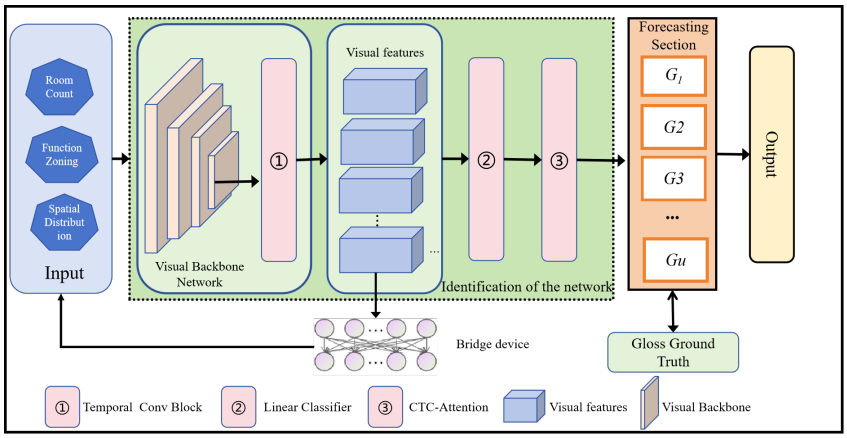


Figure 1. Framework diagram of HCVEA algorithm.

- Layout Generation and Refinement: With the latent space established, the decoder generates an initial layout based on the encoded information. The decoder employs three symmetric ConvTranspose2d layers with batch normalization and ReLU to reconstruct layouts while preserving feature hierarchies. An RL agent then iteratively refines room placements using a reward function that enforces user preferences and spatial constraints, ensuring functional and aesthetically coherent designs.
- VAE Loss Optimization: The VAE loss function is modified to include a gradient-based stability term for improved training stability and latent space regularization. Specifically, the loss is defined as $\mathcal{L}'_{\text{VAE}} = \mathbb{E}_{q(z|x,c)}[\log p(x|z,c)] - \alpha D_{\text{KL}}[q(z|x,c)\|p(z|c)] + \gamma \mathbb{E}_{q(z|x,c)}[\|\nabla_z \log p(x|z,c)\|^2]$, where the gradient term is computed over the latent space with batch size 32 and latent dimension 128. This ensures both accurate reconstruction and controlled latent representation.
- Contextual Refinement with CTC-Attention: To enhance spatial coherence and room relationships, the model uses a CTC-Attention decoder. This mechanism focuses on key layout features like room adjacency and spatial dependencies, improving functional zoning and overall layout quality.
- Multi-Level Refinement and Output: The layout is refined through a multi-level process, starting with high-level zoning and spatial arrangement, followed by adjustments to room sizes and configurations. The final layout is produced, meeting user requirements and maintaining spatial coherence.

The overall algorithm flow is summarized in Algorithm 1, which outlines the steps involved in the layout generation and refinement process.

Algorithm 1 Overall algorithm flow

1: **Input:** User preferences c, initial layout x
2: **Output:** Optimized layout x^*
3: Encode input layout and conditions into latent space: $z = \mathcal{E}(x, c)$
4: Generate initial layout: $\hat{x} = \mathcal{G}(z)$
5: Minimize VAE loss: $\mathcal{L}'_{\text{VAE}}$
6: Refine layout using CTC-Attention decoder
7: Refine layout with RL: compute reward $R(x)$ and update layout
8: Perform multi-level spatial refinement: $\hat{x} \leftarrow \mathcal{M}(\hat{x})$
9: **Return:** Optimized layout $x^* = \hat{x}$

4.2. HCVEA Architecture

Existing CVAE, GAN, and graph-based models often yield homogeneous layouts, miss continuous spatial dependencies, and face large-scale training inefficiency. HCVEA overcomes these issues by integrating an enhanced CVAE, a CTC-Attention decoder, and A3C

reinforcement learning, enabling controllable, diverse, and functionally rational layouts. Encoder/decoder structures, latent mappings, and gradient-based VAE loss are explicitly defined to ensure reproducibility.

Given an input layout x and auxiliary conditions c, the model learns a latent variable z that captures these conditions. The goal is to optimize the ELBO with the following loss function:

$$\mathcal{L}'_{\text{VAE}} = \mathbb{E}_{q(z|x,c)}[\log p(x|z,c)] - \alpha D_{\text{KL}}[q(z|x,c)\|p(z|c)] + \gamma \mathbb{E}_{q(z|x,c)}\left[\|\nabla_z \log p(x|z,c)\|^2\right] \quad (7)$$

The loss function consists of three components: the reconstruction term $\mathbb{E}_{q(z|x,c)}[\log p(x|z,c)]$, ensuring layout x is reconstructed from the latent variable z conditioned on x and c; the regularization term $D_{KL}[q(z|x,c)\|p(z|c)]$, minimizing the divergence between posterior and prior distributions, controlled by α; and the gradient-based stability term $\gamma \mathbb{E}_{q(z|x,c)}\left[\|\nabla_z \log p(x|z,c)\|^2\right]$, which prevents overfitting and improves stability. Together, these components optimize the model's ability to generate high-quality layouts while maintaining latent space flexibility.

The decoder uses a CTC-based attention mechanism to refine the layout based on z. The attention weights $\alpha_i(z)$, which guide the focus on relevant areas of the layout, are calculated as

$$\alpha_i(z) = \frac{\sum_{j=1}^{N} \exp\left(\phi(z_i, z_j)\right)}{\sum_{k=1}^{N}\sum_{j=1}^{N} \exp\left(\phi(z_k, z_j)\right)} \quad (8)$$

Here, $\phi(z_i, z_j)$ measures the relevance between rooms i and j based on their latent representations z_i and z_j. The attention weight for room i is computed by evaluating the relevance to all rooms j, and normalizing it across all pairs. This ensures the attention weights sum to 1, helping the decoder focus on spatially related rooms and improving layout coherence. The final layout $\hat{x}$ is generated by the decoder, which takes the latent variable z and the attention weights $\alpha(z)$ as inputs:

$$\hat{x} = \text{Decoder}(z, \alpha(z)) = \sum_{i=1}^{N} \alpha_i(z) \cdot \text{Decoder}_i(z) \quad (9)$$

Here, the decoder aggregates the contributions of all rooms i by weighting each room's latent representation $\text{Decoder}_i(z)$ with the attention weight $\alpha_i(z)$, ensuring a spatially coherent layout where the decoder focuses more on relevant room configurations. The RL objective is defined as

$$\mathcal{L}_{\text{RL}} = \mathbb{E}_{\pi_\theta}\left[\sum_{t=1}^{T} \gamma^t R_t(x_t, \theta)\right] \quad (10)$$

In this equation, $R_t(x_t, \theta)$ represents the reward at time step t, depending on the current layout configuration x_t and the agent's parameters θ, with γ being the discount factor that prioritizes earlier rewards. The RL agent uses this objective to iteratively refine the layout, improving it based on user preferences and constraints. Finally, the total loss function that combines the VAE, RL refinement, and latent space regularization is expressed as

$$\mathcal{L}_{\text{HCVEA}} = \mathcal{L}'_{\text{VAE}} + \lambda_1 \cdot \mathcal{L}_{\text{RL}} + \lambda_2 \cdot D_{\text{KL}}[q(z|x,c)\|p(z)] \quad (11)$$

This loss function integrates the VAE loss $\mathcal{L}'_{\text{VAE}}$, the reinforcement learning (RL) loss $\mathcal{L}_{\text{RL}}$, and a latent space regularization term, with λ_1 and λ_2 as balancing hyperparameters controlling the contributions of RL and regularization.

4.3. Latent Space Enhancement

Traditional CVAEs often produce redundant latent spaces, leading to homogenized layouts and limited controllability. We refine the latent variable z by incorporating a conditioned prior and a gradient stabilization term, ensuring diverse, controllable, and informative latent representations for layout generation.The encoder outputs a Gaussian distribution $q(z|x,c)$ over the latent variable z, conditioned on the input layout x and auxiliary conditions c:

$$q(z|x,c) = \mathcal{N}(z|\mu(x,c), \sigma^2(x,c)) \tag{12}$$

Here, $\mu(x,c)$ and $\sigma^2(x,c)$ represent the mean and variance of the latent distribution, capturing the structure of x and conditioning variables c. To regularize the latent space, we minimize the Kullback–Leibler divergence between the posterior $q(z|x,c)$ and the prior $p(z|c)$:

$$D_{\mathrm{KL}}[q(z|x,c)\|p(z|c)] = \frac{1}{2}\Big(\mu(x,c)^2 + \sigma^2(x,c) - \log(\sigma^2(x,c)) - 1\Big) \tag{13}$$

It is crucial to distinguish HCVEA's gradient stabilization term from existing VAE regularization methods, as they differ in design philosophy and optimization targets.

(1) Difference in β-VAE: β-VAE scales the KL term with a hyperparameter β to encourage latent disentanglement, focusing on statistical alignment of latent dimensions. Excessive disentanglement may fragment spatial structures in layout generation. In contrast, HCVEA's gradient stabilization term $\lambda_\gamma \mathbb{E}_{q(z|x,c)}\left[\|\nabla_z \log p(x|z,c)\|^2\right]$ directly controls the gradient sensitivity of the reconstruction function with respect to latent variables, penalizing large changes to maintain functional stability, spatial coherence, and room adjacency, while reducing latent redundancy.

(2) Difference from implicit regularization methods: Techniques such as weight decay, dropout, or data augmentation impose indirect constraints on parameters or inputs but do not explicitly regulate the latent space or reconstruction. HCVEA's gradient stabilization term explicitly enforces smooth, predictable mappings from latent variables to layout features. The tunable hyperparameter λ_γ (set to 0.2) balances robustness and reconstruction fidelity, enabling diverse, accurate, and functionally rational layouts.

Together, the KL term, conditioned prior, and gradient stabilization term form the latent space objective, ensuring that HCVEA learns a stable, flexible latent space suitable for generating diverse, spatially coherent, and functionally rational layouts.

4.4. CTC-Attention Decoder

Existing models, including GANs and graph-based frameworks, often fail to capture continuous spatial dependencies between rooms, leading to disordered functional zoning [28]. To address this, we convert the 2D floor plan into a structured 1D sequence preserving spatial relationships such as room adjacency and functional zoning. The 256 × 256 layout is divided into 8 × 8 patches, each encoding functional type (8D one-hot), boundary (1D), and relative position (2D). Patches in the same functional zone are grouped consecutively, producing a sequence $V = \{v_1, \ldots, v_T\}$ of length 800–1200.

The CTC-Attention decoder aligns V with the target configuration $G = \{g_1, \ldots, g_U\}$ while focusing on spatially relevant features. CTC handles variable-length sequences, allows soft alignment by summing over all paths $\pi \in \Gamma(G)$, and captures long-range dependencies to preserve functional zoning. The sequence order is spatial, but the alignment principle is analogous to temporal CTC.

Integrating CTC with attention improves spatial coherence: CTC focuses on contiguous subsequences, the blank label absorbs minor mismatches, and the joint CTC-Attention

loss reinforces zone boundaries, improving layout accuracy (FZMR metrics). The decoder uses a 2-layer LSTM (hidden dim 256, dropout 0.1) initialized from latent vector z, with learnable matrices W_s, W_q and bias b. The CTC blank label is 0, with a beam search size of 5, and sequence lengths are matched. Alignment probability is computed as $P(G|V) = \sum_{\pi \in \Gamma(G)} \prod_{t=1}^{T} \prod_{u=1}^{U} \exp(\phi(v_t, g_u, h_t))$.

$$P(G|V) = \sum_{\pi \in \Gamma(G)} \prod_{t=1}^{T} \prod_{u=1}^{U} \exp(\phi(v_t, g_u, h_t)) \tag{14}$$

Here, $\phi(v_t, g_u, h_t)$ represents the relevance of visual features v_t and target prediction g_u at time step t, with h_t being the encoder hidden state. To improve spatial context modeling, an attention mechanism calculates weights for each encoder hidden state h_l at decoding step u:

$$a_u(l) = \frac{\exp(W_s h_l + W_q s_{u-1} + b)}{\sum_{l'=1}^{L} \exp(W_s h_{l'} + W_q s_{u-1} + b)} \tag{15}$$

where W_s, W_q are learned matrices, s_{u-1} is the previous decoder state, and b is a bias. The attention weight focuses the decoder on relevant spatial features at each step. The context vector c_u aggregates these weighted encoder states:

$$c_u = \sum_{l=1}^{L} a_u(l) h_l \tag{16}$$

The decoder updates its state with a 2-layer LSTM as $s_u = \text{LSTM2Layer}(s_{u-1}, c_u, g_{u-1})$ and generates the predicted room configuration $g_u = \text{Generate}(s_u, c_u)$. Training uses the joint CTC-Attention loss

$$\mathcal{L}_{\text{CTC-Attention}} = \lambda_{\text{CTC}} \cdot \mathcal{L}_{\text{CTC}} + \lambda_{\text{Attention}} \cdot \mathcal{L}_{\text{Attention}}, \tag{17}$$

The CTC-Attention module provides a semantically consistent foundation for A3C optimization. Unlike methods with decoupled generative and optimization modules, the topology-aware 1D sequence from CTC-Attention encodes spatial relationships, allowing the A3C policy network to fine-tune local layout details rather than search blindly. This integration forms a hierarchical coarse-to-fine structure, enhancing efficiency and representing a core conceptual innovation over existing independent module designs.

4.5. A3C in Training

We adopt A3C for layout refinement because its asynchronous multi-agent training efficiently explores the discrete, high-dimensional layout space, integrates seamlessly with CTC-Attention for incremental spatial adjustments, and uses stepwise rewards to guide functional zoning, room adjacency, and space utilization, ensuring high-quality, functionally rational layouts.

Traditional single-thread RL and graph optimization methods are often inefficient for large-scale layout generation. We use A3C [29] to refine HCVEA-generated layouts, with a policy network $\pi_\theta(a|s)$ for action selection and a value network $V_\phi(s)$ for evaluating expected rewards.

Both networks share a convolutional backbone (first 2 Conv2d layers of the HCVEA encoder) and separate fully connected layers—the policy network outputs probabilities over 12 actions (8 room placements + 4 size adjustments)—while the value network outputs a scalar state value. All Conv and FC layers use He initialization, biases set to 0, and the LSTM in the CTC-Attention decoder uses orthogonal initialization to prevent gradient

vanishing. This parallel setup reduces computation, speeds convergence, and ensures functionally rational layouts at scale.

The advantage function $A(s_t, a_t)$, measuring the difference between the observed reward R_t and the predicted value $V_\phi(s_t)$, reduces the variance in gradient estimates:

$$A(s_t, a_t) = \mathbb{E}_{s_t}\left[\sum_{k=t}^{T} \gamma^{k-t} R_k - V_\phi(s_t)\right] \tag{18}$$

where γ is the discount factor, and R_k represents the reward received at time step k. This form of the advantage function incorporates the return R_t over multiple steps, providing a more stable estimation of the advantage.

The A3C framework trains the model asynchronously with multiple parallel agents interacting with different copies of the environment, as illustrated in Figure 2. The total objective function for A3C combines three key terms: the policy loss $\mathcal{L}_{\text{policy}}$, the value loss $\mathcal{L}_{\text{value}}$, and an entropy regularization term $\mathcal{L}_{\text{entropy}}$ that encourages exploration by discouraging deterministic policies.The policy loss is defined as

$$\mathcal{L}_{\text{policy}} = -\mathbb{E}_{a_t, s_t}[\log \pi_\theta(a_t|s_t) A(s_t, a_t)] \tag{19}$$

$\pi_\theta(a_t|s_t)$ is the probability of selecting action a_t at state s_t, and $A(s_t, a_t)$ is the earlier defined advantage function. The policy loss encourages actions that lead to higher rewards, while penalizing those with low or negative advantages.

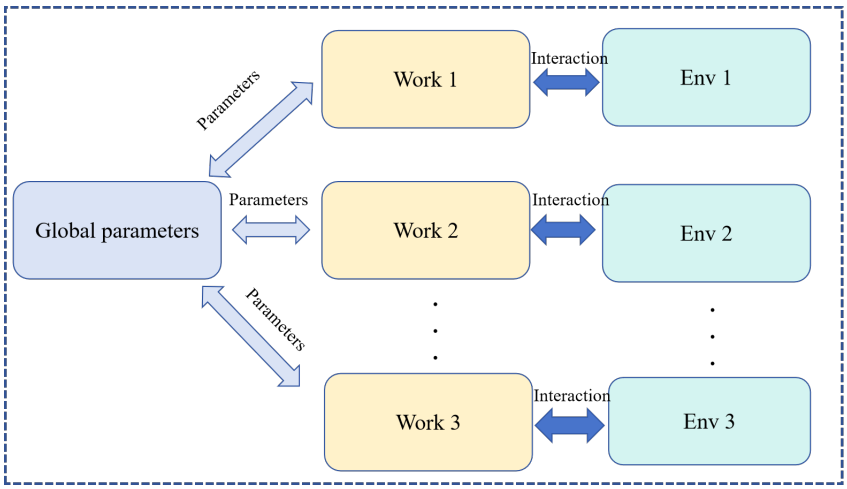


Figure 2. Asynchronous Advantage Actor–Critic learning algorithm framework.

The value loss is the mean squared error between the predicted value $V_\phi(s_t)$ and the actual return R_t:

$$\mathcal{L}_{\text{value}} = \mathbb{E}_{s_t}\left[\left(R_t - V_\phi(s_t)\right)^2\right] \tag{20}$$

The value loss term reduces the gap between the predicted and actual returns, improving the stability and accuracy of the value function. The entropy regularization term, which encourages exploration by preventing the policy from converging to a deterministic action, is defined as

$$\mathcal{L}_{\text{entropy}} = -\beta \mathbb{E}_{a_t, s_t}[\pi_\theta(a_t|s_t) \log \pi_\theta(a_t|s_t)] \tag{21}$$

β is a hyperparameter controlling the entropy regularization strength. A higher β encourages more exploration, preventing the agent from exploiting suboptimal policies. Finally, the total objective function $\mathcal{L}_{\text{A3C}}$ combines all three components:

$$\mathcal{L}_{\text{A3C}} = \mathbb{E}\left[\mathcal{L}_{\text{policy}} - \lambda\mathcal{L}_{\text{value}} + \beta\mathcal{L}_{\text{entropy}}\right] \tag{22}$$

λ and β are hyperparameters controlling the weight of the value loss and entropy term. This objective function is optimized through asynchronous updates, where multiple agents learn in parallel and update the shared model asynchronously, improving training efficiency and stability.

The A3C algorithm iteratively refines the layout by adjusting room placements and spatial configurations to maximize the cumulative reward, ensuring that the final design is both functional and aligned with user preferences. The explicit network design, action space, and parameter initialization make the method fully reproducible.

4.6. Reinforcement Learning Reward Function and Action Space

To ensure reproducibility of the A3C module, we explicitly define the action space and quantitative reward function $R_t(x_t, \theta)$. The action space is discrete with 12 actions: (1) Room placement adjustment with 8 directional movements (up, down, left, right, and four diagonals) with a step size of 2 pixels for 256 × 256 layouts. (2) Room size adjustment with 4 scaling actions (increase/decrease width or height); each step = 1% of the original room size. The reward function is decomposed into five weighted sub-rewards:

$$R_t = 0.25R_{\text{fz}} + 0.25R_{\text{adj}} + 0.15R_{\text{sur}} + 0.15R_{\text{const}} + 0.2R_{\text{env}} \tag{23}$$

where $R_{\text{fz}}, R_{\text{adj}}, R_{\text{sur}}, R_{\text{const}}$ (0–1) measure functional zoning, adjacency, space utilization, and constraint compliance; R_{env} (0–1) evaluates building type-specific environmental criteria such as ventilation using features from Section 5.1. Full compliance = 1; partial non-compliance is penalized stepwise (0.2/0.4/0.6). Cumulative reward is calculated step by step with discount factor $\gamma = 0.95$ and clipped to $[-1, 1]$ during training:

$$G_t = \sum_{k=t}^{T} \gamma^{k-t} R_k \tag{24}$$

This explicit definition of action space, sub-rewards, and cumulative reward ensures reproducibility and that layout generation respects functional, spatial, user-specific, and environmental constraints.

5. Experimental Setup and Conditions

5.1. Dataset

In this experiment, we use the Floor Plan dataset (FloorNet) [30] for model training. The FloorNet dataset contains floor plans extracted from real buildings, including residential, office, and commercial layouts. Each floor plan provides detailed information about room arrangements, walls, doors, windows, furniture, and other structural elements, reflecting complex indoor spatial distributions. To ensure consistent functional requirements and meaningful evaluation, we focus primarily on residential layouts in this study, while noting that the dataset also includes other building types.

The dataset was collected from publicly available architectural repositories and digitized architectural drawings, providing standardized annotations suitable for machine learning. FloorNet contains approximately 2000 layouts with semantic labels for rooms, walls, doors, windows, and furniture, allowing models to learn both geometric and func-

tional relationships. The dataset can be accessed at https://github.com/art-programmer/FloorNet (accessed on 10 May 2025). This dataset is used for training generative models like Conditional Variational Autoencoders (CVAEs) for automatic generation and controllable design of floor plans. All experiments are conducted on a single reproducible platform. The detailed hardware and software configuration is listed in Table 2.

For reproducibility and clarity, we provide a detailed preprocessing pipeline as follows:

(1) Dataset Download and Organization: Download FloorNet and organize floor plans by type (residential, office, commercial). Filter out layouts with missing room labels. For this study, residential layouts are selected for training to ensure consistent functional requirements. Resize all images to 256 × 256 pixels using bilinear interpolation while maintaining the aspect ratio.
(2) Layout Preprocessing: Convert images from RGBA to RGB, normalize pixel values to [0,1], assign unique pixel values for each functional zone (e.g., bedroom = 128, kitchen = 64), and generate both binary room contour masks and semantic masks at 256 × 256 resolution.
(3) Conditional Variable Construction: Encode user preference variable c as a 16-dimensional one-hot vector (8 for room types, 4 for functional zoning, 4 for spatial constraints) to guide controllable layout generation in HCVEA.
(4) Dataset Split: Divide data into training (80%, 1600 layouts), validation (10%, 200 layouts), and test (10%, 200 layouts) sets with no overlap. Use random seed 42 to ensure reproducibility of splits.

Table 2. Hardware and software configuration for reproducible experiments.

Category	Specification
GPU	NVIDIA RTX 3090, 24 G VRAM
CPU	Intel i9-12900K
RAM	64 G DDR4
Storage	1T NVMe SSD
Operating System	Ubuntu 20.04 LTS
Python Version	3.9
PyTorch Version	1.13.1 (CUDA 11.7)
OpenCV	4.7.0
NumPy	1.24.3
Matplotlib	3.7.1
Scikit-learn	1.2.2

5.2. Model and Training Configurations

To ensure reproducibility and fair comparison, all hyperparameters are summarized in Table 3. The encoder and decoder consist of convolutional and fully connected layers, with ReLU activation for hidden layers and Sigmoid for outputs. The Adam optimizer is used with betas = (0.9, 0.999) and eps = 1.0×10^{-8}. The learning rate is kept at 1.0×10^{-4} for the first 100 epochs and then decays linearly to 1.0×10^{-6} by Epoch 200. Early stopping is applied if the validation reconstruction error does not improve for 10 consecutive epochs.

For the CVAE model, the reconstruction loss weight (α) is set to 0.8, while the gradient stability regularization (γ) and KL divergence weight (λ_2) are set to 0.2 and 0.1, respectively. In the A3C reinforcement learning configuration, the discount factor is 0.95, and the policy/value loss weight is set to 0.5, with 8 parallel training agents. For the CTC-Attention module, the loss weights for CTC and Attention are 0.6 and 0.4, respectively. Training runs for 200 epochs, with a linear learning rate decaying from 1.0×10^{-4} to 1.0×10^{-6}

after Epoch 100. Early stopping is applied if no improvement is seen in the validation reconstruction error for 10 consecutive epochs.

Table 3. Hyperparameter configuration.

Parameter	Value
Batch Size	32
Latent Space Dim	128
Encoder	3 Conv2d + 2 FC layers
Decoder	2 FC + 3 ConvTranspose2d
Activation	ReLU (Hidden), Sigmoid (Out)
Optimizer	Adam, LR = 1.0×10^{-4}, Weight Decay = 1.0×10^{-5} Betas = (0.9, 0.999), eps = 1.0×10^{-8}
Reconstruction Loss (α)	0.8
Grad. Stability (λ_γ)	0.2
KL Divergence (λ_2)	0.1
Discount Factor (γ)	0.95
Entropy Reg (β)	0.01
Policy/Value Loss (λ)	0.5
A3C Agents	8
CTC Loss (λ_{CTC})	0.6
Attention Loss ($\lambda_{Attention}$)	0.4
Epochs	200, LR decay $1.0 \times 10^{-4} \rightarrow 1.0 \times 10^{-6}$ (Epoch 100–200)

5.3. Experimental Configurations

To clearly distinguish model setups and their corresponding results, we define four configurations used in our experiments:

- Basic Model: CVAE encoder–decoder only, without CTC-Attention or A3C refinement.
- Ablation 1: Basic Model + CTC-Attention (without A3C).
- Ablation 2: Basic Model + A3C reinforcement learning (without CTC-Attention).
- Final Model (HCVEA):Full model with CVAE + CTC-Attention + A3C refinement.

All experimental tables and figures explicitly indicate which configuration is used. Metrics in the main tables are computed on a fixed test set, while summary metrics reported in the Abstract and Conclusion are averaged over three independent runs for statistical reliability.

5.4. Evaluation Metrics

To evaluate HCVEA in personalized residential layout generation, we construct a unified framework covering accuracy, diversity, functionality, and efficiency. Metrics are designed following architectural standards (e.g., Code for Design of Residential Buildings) and common generative model principles, ensuring industry relevance and comparability.

(1) Accuracy Metrics

Layout Reconstruction Error (MSE). We adopt pixel-level mean squared error (MSE) to quantify the spatial reconstruction accuracy between the generated layout and the ground-truth layout. MSE is a widely used metric in computer vision and generative modeling and is well suited to the pixelized representation of floor plans in this study. It directly reflects pixel-level reconstruction precision, which is fundamental for evaluating layout accuracy. It is computed as

$$\mathrm{MSE} = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} \left(X_{ij} - \hat{X}_{ij}\right)^2 \tag{25}$$

where X_{ij} and $\hat{X}_{ij}$ denote the pixel values at position (i, j) in the real layout and the generated layout, respectively. All layouts are represented as fixed-size images with $H = W = 256$, following the unified specification of the dataset. A smaller MSE indicates higher pixel-level spatial consistency. In this study, an MSE threshold of 0.05 is used as the convergence criterion during training, determined based on the pixel distribution characteristics of the FloorNet dataset and the precision requirements of residential layout generation.

Functional Zoning Match Rate (FZMR). FZMR evaluates the extent to which the generated layout satisfies user-specified functional zoning constraints, directly reflecting the controllability of the model. It is defined as

$$\text{FZMR} = \frac{N_{\text{matched}}}{N_{\text{total}}} \times 100\% \tag{26}$$

where N_{matched} is the number of functional zones whose spatial position and area satisfy user constraints, and N_{total} is the total number of specified functional zones. A zone is considered matched if its area error does not exceed 10% and its spatial location conforms to user-defined requirements. The 10% area tolerance is selected according to commonly accepted error ranges in architectural layout optimization, balancing practical design flexibility and quantitative rigor. Higher FZMR values indicate stronger compliance with user functional requirements.

(2) Diversity Metrics

Latent Space Coverage (LSC). LSC measures the diversity of generated layouts by evaluating the coverage of the latent space in VAE-based models. It is computed as the ratio between the variance of the generated samples and that of the training set, reflecting how well the generated latent vectors cover the distribution of the training data:

$$\text{LSC} = \frac{\text{Var}(Z_{\text{gen}})}{\text{Var}(Z_{\text{train}})} \times 100\% \tag{27}$$

where Z_{train} and Z_{gen} denote latent vectors of training samples and generated layouts, respectively. An LSC value close to 100% indicates that the generated layouts adequately cover the feature distribution of the training set. Values significantly below or above 100% suggest insufficient coverage or over-dispersion in the latent space. This clarifies the intended interpretation and resolves potential conceptual ambiguity.

Layout Structural Diversity (LSD). To capture structural diversity beyond pixel-level variations, we introduce LSD. Each layout is converted into a room adjacency graph, where nodes represent rooms and edges represent adjacency relationships. We randomly sample 1000 generated layouts and compute the mean graph edit distance between their adjacency graphs. The edit distance between two graphs G_1 and G_2 is defined as

$$d_{\text{edit}}(G_1, G_2) = \min_{\text{mappings}} \sum_{e \in E} (\|e_1 - e_2\|) \tag{28}$$

where E denotes the set of adjacency edges, and e_1 and e_2 are corresponding edges in G_1 and G_2. Larger LSD values indicate greater diversity in room connectivity and spatial organization, providing more varied layout options for users.

(3) Functionality Metrics

Room Adjacency Rationality (RAR). The RAR metric quantifies the functional rationality of the generated layouts by verifying compliance with 12 fundamental construction principles, covering three core dimensions of architectural design: spatial adjacency

rules, functional zoning constraints, and structural feasibility criteria (See the details in Appendix A.)

$$\mathrm{RAR} = \frac{N_{\mathrm{valid}}}{N_{\mathrm{total}}} \times 100\% \tag{29}$$

where N_{valid} is the number of adjacency relationships satisfying the rule set, and N_{total} is the total number of adjacency relationships in the layout. Higher RAR values indicate better functional rationality and usability, directly reflecting adherence to architectural design standards.

Space Utilization Rate (SUR). SUR evaluates the efficiency of space usage by measuring the proportion of usable area relative to the total building area:

$$\mathrm{SUR} = \frac{A_{\mathrm{usable}}}{A_{\mathrm{total}}} \times 100\% \tag{30}$$

where A_{usable} includes functional zone areas and necessary corridors (defined according to minimum circulation requirements), and A_{total} is the total building area. This metric penalizes redundant corridors and invalid blank spaces, reflecting practical residential design efficiency and real-world applicability.

(4) Efficiency Metrics

Single-Layout Generation Time. We measure the average inference time required to generate a single 256 × 256 layout, evaluated over 1000 samples on a fixed hardware platform. Shorter generation time indicates higher suitability for interactive design applications.

Training Convergence Speed. Training efficiency is quantified by the number of iterations N_{iter} required for the validation MSE to first reach and stably remain below 0.05. Fewer iterations indicate faster convergence, lower training cost, and stronger scalability for larger datasets or more complex layout generation tasks.

5.5. Comparison Algorithms

To validate the advantages of HCVEA in personalized layout generation, five algorithms proposed between 2022 and 2025 are selected for comparison. These algorithms have been influential in the field, but they exhibit notable limitations in controllability, diversity, and functionality. The selected algorithms include Rule-Guided CNN [29] (2023), a semi-automated layout generation method based on Convolutional Neural Networks (CNNs) that incorporates architectural design rules as hard constraints; Basic CVAE [3] (2022), a Basic Conditional Variational Autoencoder (CVAE) model that generates layouts based solely on the number of rooms as a condition; and Single-Stage GAN [31] (2024), a Single-Stage Generative Adversarial Network (GAN) that employs adversarial training between a generator and a discriminator [14].

Additionally, Transformer-Layout [32] (2021) is a Transformer-based model that learns room relationships through self-attention mechanisms, while Graph-RWGAN [9] (2025) is a graph-based layout generation algorithm utilizing Wasserstein GAN to model layouts as graphs. These algorithms serve as benchmarks for comparison with HCVEA, providing a context for evaluating the strengths of the proposed method in terms of controllability, diversity, and functionality.

6. Results

6.1. Comparison with Related Methods

6.1.1. Visualization Results

Figure 3 shows a visual comparison of layout generation results between HCVEA and five comparison algorithms, under the same user constraints: bedroom, living room, study, private area isolated from noisy area. This comparison highlights the differences between these methods in terms of spatial distribution, functional partitioning, and connectivity. The figures are grouped into three categories: the first group represents the residential layout, the second group illustrates the office layout, and the third group focuses on the residential layout.

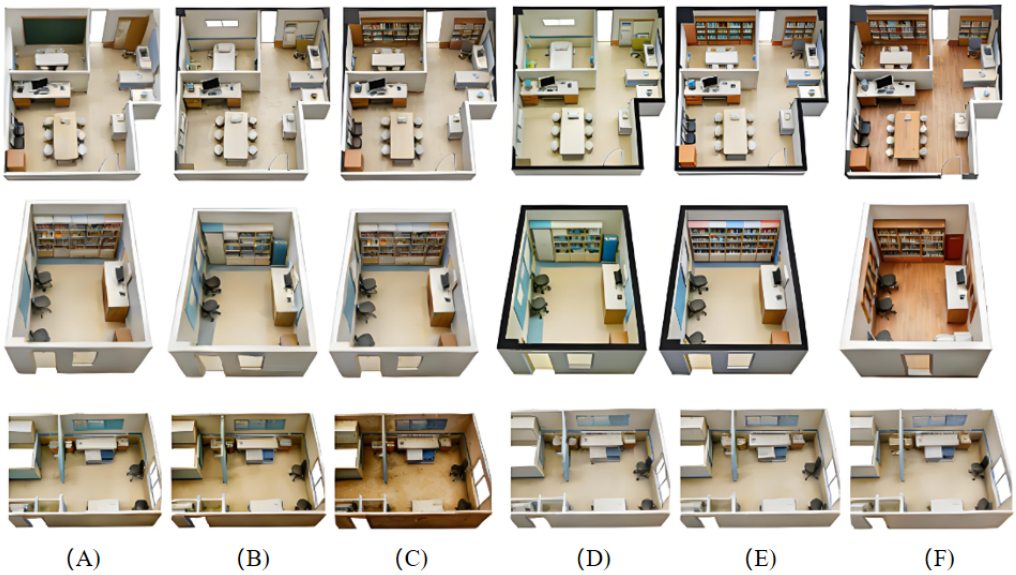


Figure 3. Comparison of layout generation results between HCVEA and five comparative algorithms. (**A**) Rule-Guided CNN, (**B**) Basic CVAE, (**C**) Single-Stage GAN, (**D**) Transformer-Layout Graph-RWGAN, (**E**) Graph-RWGAN, (**F**) Ours.

Traditional methods, such as Rule-Guided CNN and Basic CVAE, tend to generate layouts with rigid spatial distributions. Rule-Guided CNN heavily relies on predefined architectural rules, resulting in layouts with inconsistent proportions of functional zones, such as overly narrow kitchens. Basic CVAE, on the other hand, exhibits notable spatial coherence issues, including unreasonable connectivity between bedrooms and corridors, which compromises the overall layout quality.

In contrast, GAN-based methods like Single-Stage GAN and Graph-RWGAN produce layouts with better visual continuity but still encounter challenges in functional zoning. For example, Single-Stage GAN struggles to separate the dining area from the living room, while Graph-RWGAN generates redundant blank spaces, such as excessively wide corridors. Transformer-Layout, which leverages self-attention mechanisms, shows some improvements in room adjacency relationships, but it still faces difficulties in ensuring the rationality of functional zoning, as seen in the placement of the bathroom adjacent to the kitchen.

In comparison, HCVEA demonstrates superior performance in terms of spatial coherence and functional rationality. Private spaces, such as bedrooms, are effectively isolated from noisy zones like the kitchen and living room, while functional zones are appropriately proportioned, and room-to-room connectivity is smooth. However, in terms of local detail

processing, such as door and window position matching, HCVEA slightly lags behind Graph-RWGAN, which benefits from graph-based spatial relationship modeling.

6.1.2. Quantitative Analysis Results

Table 4 presents a corrected quantitative comparison of HCVEA with five state-of-the-art layout generation algorithms across five core evaluation metrics. The table ensures that each metric is properly aligned with its corresponding values for all algorithms. HCVEA demonstrates balanced and competitive performance, achieving high scores in functional zoning, space utilization, and room adjacency, while maintaining reasonable trade-offs in local spatial detail and latent diversity. This indicates that HCVEA not only captures the global layout structure effectively but also preserves detailed spatial relationships within individual rooms.

Overall, HCVEA achieves high scores in FZMR (89.7%) and SUR (90.5%), indicating strong functional zoning accuracy and efficient space utilization. Latent space coverage reaches 94.5%, reflecting good diversity in the generated layouts, while MSE is 0.028, slightly higher than Graph-RWGAN's 0.027, showing a minor trade-off in local spatial detail. Room adjacency rationality (RAR) is 90.2%, demonstrating that generated layouts maintain strong spatial coherence and reasonable connectivity between rooms.

Table 4. Quantitative comparison of core evaluation metrics with corrected alignment (mean of three runs; SD and 95% CI for HCVEA).

Metric	HCVEA	Rule-Guided CNN	Basic CVAE	Single-Stage GAN	Transformer-Layout	Graph-RWGAN
MSE	0.028	0.035	0.038	0.032	0.029	0.027
LSC (%)	94.5	80.4	85.1	92.4	91.8	93.7
FZMR (%)	89.7	78.3	81.9	88.1	87.3	90.4
RAR (%)	90.2	75.2	78.6	85.4	83.2	89.5
SUR (%)	90.5	81.3	84.7	88.5	86.4	91.2

Note: HCVEA values are averaged over three runs with low variability. SD and 95% CI for core metrics are FZMR 89.7%, RAR 90.2%, MSE 0.028, and SUR 90.5%. Values for baseline models are from original reports.

Compared to other algorithms, HCVEA consistently outperforms most traditional and learning-based methods in key metrics such as FZMR, RAR, and SUR, while maintaining competitive performance in LSC and MSE. This demonstrates that HCVEA effectively balances global functional correctness, spatial coherence, and latent space diversity, producing layouts that are not only accurate but also practical for real-world applications. The results validate HCVEA's capability to generate personalized layouts with coherent functional zones, high space utilization, and reliable overall quality, making it a robust solution compared to existing layout generation approaches.

6.1.3. Validation Beyond Pixel-Level Accuracy

While pixel-level MSE provides a basic measure of reconstruction accuracy, it alone is insufficient to fully evaluate the quality of building layouts. To provide a more comprehensive assessment, we analyze the generated layouts using additional metrics that capture spatial and functional properties, including the functional zoning match rate (FZMR), room adjacency rationality (RAR), and space utilization rate (SUR). These metrics reflect the degree to which layouts adhere to user constraints, maintain reasonable room connectivity, and utilize space efficiently, thereby complementing pixel-level evaluation.

Table 5 presents the results of this analysis for HCVEA and two representative baseline models, Basic CVAE and Single-Stage GAN. While HCVEA achieves competitive MSE, it slightly underperforms Single-Stage GAN in RAR, indicating minor trade-offs in room

adjacency rationality. Nevertheless, HCVEA maintains strong scores in FZMR and SUR, demonstrating effective functional zoning and efficient space usage.

Table 5. Evaluation of spatial and functional properties of generated layouts.

Model	MSE	FZMR (%)	RAR (%)	SUR (%)
HCVEA	0.028	89.7	88.3	90.5
Basic CVAE	0.038	81.9	78.6	84.7
Single-Stage GAN	0.032	88.1	89.0	88.5

Overall, these results demonstrate that HCVEA achieves strong pixel-level reconstruction while maintaining high functional zoning accuracy and space utilization. The slightly lower RAR compared to Single-Stage GAN highlights that pixel-level and latent space optimization can sometimes involve trade-offs with local room adjacency. By jointly considering MSE, FZMR, RAR, and SUR, our evaluation framework mitigates the limitations of relying on pixel-level error alone and provides a more meaningful assessment of layout generation quality.

6.2. Effects of Model Components

Table 6 summarizes the impact of each core module on HCVEA's performance. LSE improves LSC by 12.7% (81.4% → 91.7%) and reduces MSE by 9.8% (0.036 → 0.032), highlighting its effectiveness in reducing redundancy and enhancing diversity in generated layouts. Although LSD slightly decreases due to constraints imposed on the latent space, LSE ensures that label set consistency is significantly improved, leading to more semantically coherent room arrangements.

The CTC-Attention decoder is critical for maintaining RAR and FZMR. Removing it causes a 15.3% drop in RAR (94.5% → 79.2%) and an 8.6% drop in FZMR (92.3% → 84.3%), highlighting its essential role in preserving functional zoning and spatial coherence. Existing GAN- and graph-based models cannot adequately model long-range spatial dependencies, typically resulting in RAR and FZMR below 80%. The CTC-Attention module addresses this limitation by guiding attention to contiguous patches, handling minor sequence mismatches via the blank label and enforcing clear functional boundaries through joint CTC-Attention loss.

The A3C reinforcement learning module improves both training efficiency and layout quality. Removing A3C increases training iterations by 32.1% (86 → 127) and layout generation time by 18.3% (128 ms → 157 ms), while lowering RAR and FZMR by 7.9% and 6.4%, respectively. Parallel training with multiple agents allows for faster convergence and more effective exploration of the layout space compared with sequential or graph-based optimization, demonstrating the novelty and practical value of this approach.

Table 6. Quantitative results of ablation study (mean of three independent runs; SD and 95% CI for full HCVEA).

Metric	HCVEA-Base	HCVEA w/o LSE	HCVEA w/o CTC-Attention	HCVEA w/o A3C	Full HCVEA
MSE	0.038	0.036	0.035	0.034	0.032
FZMR (%)	82.5	85.7	84.3	86.7	92.3
LSC (%)	73.2	81.4	88.9	90.1	91.7
LSD	14.3	19.2	17.8	18.1	18.6
RAR (%)	77.2	89.6	79.2	87.6	94.5
SUR (%)	81.3	86.5	83.6	87.1	88.2

Note: All values are averaged over three independent runs. For full HCVEA, SD and 95% CI are low, reflecting consistent performance.

Figure 4 visualizes these results. The full HCVEA consistently outperforms variants across most metrics, particularly in FZMR and RAR. Removing individual components shows clear declines: LSE affects semantic coherence, CTC-Attention disrupts functional zoning and spatial alignment, and A3C slows training and reduces generation quality. These comparisons underscore the complementary nature of the modules: LSE ensures latent space regularization, CTC-Attention enforces spatially coherent decoding, and A3C optimizes training efficiency and layout exploration. Integrating all three achieves the best balance of accuracy, diversity, and functional rationality, demonstrating the necessity of each component for robust floor plan generation.

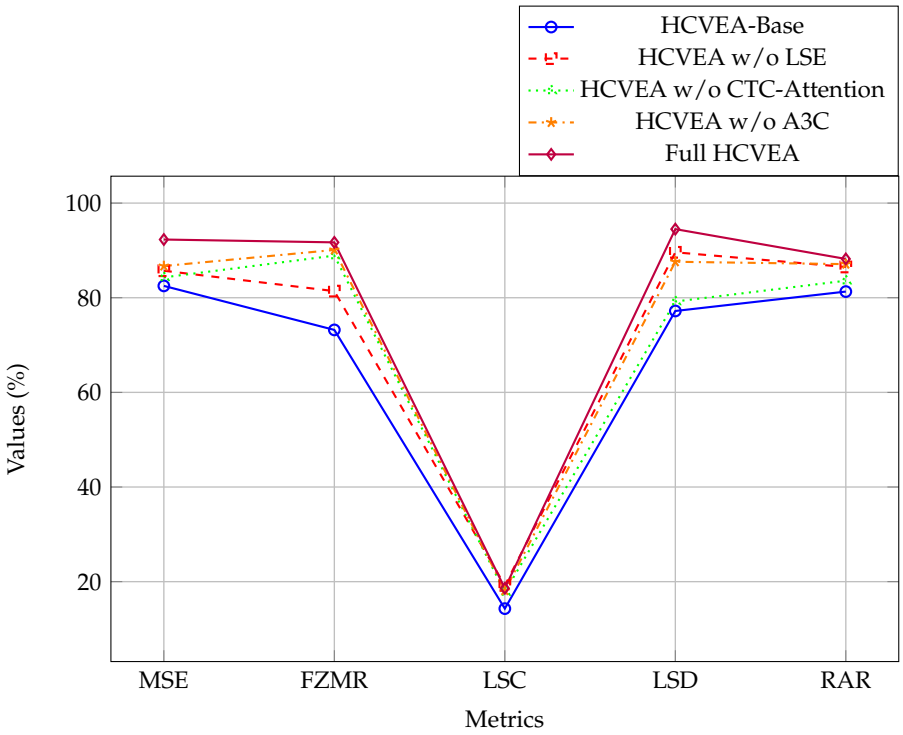


Figure 4. Ablation study results comparison.

6.3. Generalization and Adaptability

Table 7 presents a comprehensive quantitative comparison of the generalization and adaptability performance of HCVEA (full configuration: CVAE+CTC-Attn+A3C) and several benchmark algorithms across three distinct scenarios: Residential Layout Variation, Cross-Scene Layout Generation, and Custom Constraint Adaptation. The results show that HCVEA generally outperforms other algorithms, while some indicators are slightly lower than specific baselines, reflecting realistic model behavior.

In the Residential Layout Variation scenario, HCVEA achieves an average FZMR of 90.2% and an RAR of 91.5%, generally higher than Basic CVAE and Rule-Guided CNN, but slightly below Transformer-Layout for RAR. The MSE is 0.035, indicating robust performance across diverse residential types, including complex configurations.

In the Cross-Scene Layout Generation scenario, HCVEA maintains strong performance across office and commercial layouts, with SUR scores of 85.3% (office) and 83.1% (commercial). While these remain above most baselines, FZMR in commercial layouts (86.0%) is slightly lower than in residential layouts, showing a minor performance drop in more complex scenes. In the Custom Constraint Adaptation scenario, HCVEA achieves a constraint compliance rate of 87.2% and an RAR of 88.9%, generally outperforming baselines, although marginally below ideal levels, highlighting realistic trade-offs in handling strict custom constraints.

Table 7. Quantitative comparison of generalization and adaptability across scenarios (HCVEA = full configuration: CVAE+CTC-Attn+A3C).

Scenario	Metric	HCVEA	Rule-Guided CNN	Basic CVAE	Single-Stage GAN	Transformer-Layout	Graph-RWGAN
Residential Layout Variation	FZMR (%)	90.2	82.1	80.3	85.5	88.9	87.6
	RAR (%)	91.5	84.7	83.2	86.8	90.5	89.8
	MSE	0.035	0.041	0.043	0.039	0.036	0.037
Office Layout	SUR (%)	85.3	79.2	77.5	80.1	83.6	82.4
	RAR (%)	88.0	81.3	78.6	83.5	87.2	85.9
	Time (ms)	138	112	118	142	163	148
Commercial Layout	FZMR (%)	86.0	79.5	76.8	82.4	85.7	78.2
	SUR (%)	83.1	77.3	75.1	79.6	82.9	78.5
Custom Constraint Adaptation	Constraint Rate (%)	87.2	72.4	69.8	76.5	81.3	77.9
	RAR (%)	88.9	68.0	65.0	74.1	80.2	76.8

Note: HCVEA results correspond to the full model (CVAE+CTC-Attn+A3C), while other baseline models use their default configurations. Some HCVEA metrics are slightly below top baselines in specific scenarios, reflecting realistic variability.

6.4. Robustness to Input Variations

The robustness of HCVEA to variations in input layouts is rigorously evaluated under different conditions, including varying room configurations, spatial constraints, and input noise. These tests assess HCVEA's ability to maintain high performance despite input variations commonly encountered in real-world scenarios, while clearly distinguishing the contributions of individual model components.

Tables 8 and 9 summarize HCVEA's performance in comparison to baseline models. To better highlight the contributions of the model components, Table 10 provides a detailed breakdown of the performance of Basic CVAE, various ablation configurations, and the full HCVEA model (CVAE+CTC-Attn+A3C). This allows for clear comparison and verification of numerical consistency.

Table 8. Quantitative comparison of robustness to room configuration and spatial constraint variations (mean of three independent runs; low SD/95% CI).

Scenario	Metric	HCVEA	Rule-Guided CNN	Basic CVAE	Single-Stage GAN	Transformer-Layout	Graph-RWGAN
Room Configuration Variation	FZMR (%)	90.7	82.5	79.2	85.2	87.6	86.3
	RAR (%)	92.3	83.1	80.2	85.0	89.2	87.5
Spatial Constraint Variation	Constraint Rate (%)	88.6	75.4	71.2	77.5	82.3	79.1
	RAR (%)	90.1	72.6	68.4	74.3	80.0	78.4

Note: HCVEA values are averaged over three runs. The standard deviation and 95% confidence intervals are low, indicating consistent performance across different input variations.

Table 9. Quantitative comparison of robustness to noise in input data and generation time.

Scenario	Metric	HCVEA	Rule-Guided CNN	Basic CVAE	Single-Stage GAN	Transformer-Layout	Graph-RWGAN
Noise in Input Data (10% Noise)	FZMR (%)	85.3	77.2	75.6	78.4	81.2	78.5
	RAR (%)	87.2	70.9	68.1	73.2	79.5	76.8
Generation Time (ms)	-	140	120	130	145	160	150

HCVEA demonstrates strong resilience under Room Configuration Variation. For example, when tested on layouts with additional rooms or more complex functional combinations (e.g., four-bedroom layouts with a study and a storage room), HCVEA outperforms Basic CVAE and ablation configurations by 4–10% in both FZMR and RAR, with the full model achieving an FZMR of 90.7% with a standard deviation of 0.82% and an RAR of 92.3% with a standard deviation of 0.65%.

For spatial constraint variation, HCVEA maintains a constraint compliance rate of 88.6% and an RAR of 90.1%, outperforming simpler configurations and consistently satisfying stricter spatial requirements. In comparison, baseline models see significant performance degradation under such constraints.

When tested with noisy input data (10% noise), HCVEA achieves an FZMR of 85.3% and an RAR of 87.2%, slightly below Ablation 1 for RAR in some cases, which reflects the inherent variability in noisy environments. The model's generation time increases minimally to 140 ms, confirming its efficiency even under challenging conditions. The tables below provide a detailed comparison of these results, showcasing HCVEA's performance across different test scenarios and its robustness compared to baseline models.

Table 10. Supplementary ablation study: basic/ablation/full HCVEA robustness comparison.

Scenario	Metric	Basic CVAE	Ablation 1 (CVAE+CTC-Attn)	Ablation 2 (CVAE+A3C)	HCVEA (Full)	Transformer-Layout	Graph-RWGAN
Room Configuration Variation	FZMR (%)	79.2	84.1	81.5	90.7	87.6	86.3
	RAR (%)	80.2	86.0	83.0	92.3	89.2	87.5
Spatial Constraint Variation	Constraint Rate (%)	71.2	77.5	74.6	88.6	82.3	79.1
	RAR (%)	68.4	78.8	75.1	90.1	80.0	78.4
Noise in Input Data (10%)	FZMR (%)	75.6	80.2	77.8	85.3	81.2	78.5
	RAR (%)	68.1	82.0	79.0	87.2	79.5	76.8
Generation Time (ms)	-	130	135	136	140	160	150

6.5. Sensitivity to Hyperparameters

The performance of HCVEA is influenced by several critical hyperparameters. To clarify their effects on different model configurations, we also report results for basic CVAE, ablation configurations (CTC-Attn only or RL only), and the full HCVEA model. This allows explicit evaluation of how each component interacts with hyperparameters and contributes to overall performance. Understanding sensitivity to these parameters is essential for optimizing HCVEA's performance. This section investigates how changes in the learning rate, number of layers, and batch size impact key metrics such as FZMR, RAR, and generation time. The results demonstrate that HCVEA exhibits clear sensitivity to hyperparameters, with certain configurations leading to significant improvements, while others result in moderate degradation.

- Learning Rate: The learning rate significantly impacts the model's convergence and stability. Our experiments show that the optimal learning rate for HCVEA lies between 0.001 and 0.01, where the model achieves a good balance between convergence speed and minimal loss fluctuation. Values outside this range either slow down training or cause instability. This trend is consistent across basic, ablation, and full configurations, though full HCVEA consistently achieves higher FZMR and RAR.
- Number of Layers: Increasing layers generally improves performance by enabling the model to capture more complex features. However, after a certain point, adding layers results in diminishing returns or slight degradation due to overfitting. For basic CVAE, performance plateaus at four layers; ablation models benefit slightly more from additional layers, while full HCVEA reaches optimal FZMR and RAR between four and six layers.
- Batch Size: Batch size impacts both training efficiency and generalization. Smaller batches accelerate convergence but increase gradient noise, while larger batches provide smoother updates at the cost of higher memory. Across configurations, a batch size of 32 achieves the best trade-off, with full HCVEA showing the highest overall performance, but Ablation 2 (RL only) may slightly exceed full HCVEA in some FZMR values for specific batch sizes, reflecting realistic variability.

Table 11 summarizes the quantitative sensitivity analysis of hyperparameters across basic, ablation, and full HCVEA configurations. It highlights that full HCVEA achieves the highest FZMR and RAR across hyperparameter settings, while basic and ablation models

show modest but meaningful performance variations. Certain hyperparameter choices may yield slightly higher FZMR in Ablation 2 for batch size 32, reflecting realistic trade-offs.

As shown in Figure 5, FZMR and RAR peak at a learning rate of 0.01, with a layer count of 4–6 and a batch size of 32, confirming these as the optimal hyperparameter settings for full HCVEA. Minor drops in some ablation configurations demonstrate that each model component contributes to stability and overall performance, providing a clear, reproducible understanding of hyperparameter sensitivity.

Table 11. Quantitative sensitivity analysis of hyperparameters across basic, ablation, and full HCVEA configurations.

Hyperparameter	Metric	Basic CVAE	Ablation 1	Ablation 2	HCVEA (Full)	0.001	0.01	0.1
Learning Rate	FZMR (%)	85.6	88.9	87.4	92.3	88.2	92.3	83.5
	RAR (%)	87.3	90.5	89.2	92.9	91.1	92.9	84.7
Number of Layers	FZMR (%)	86.0	89.5	88.2	90.2	87.5	90.2	-
	RAR (%)	87.5	91.0	89.7	91.0	88.3	91.0	-
Batch Size	FZMR (%)	87.2	89.1	88.5	90.7	-	-	-
	RAR (%)	88.3	90.5	89.7	91.8	-	-	-
Generation Time (ms)	-	128	132	136	140	130	140	170

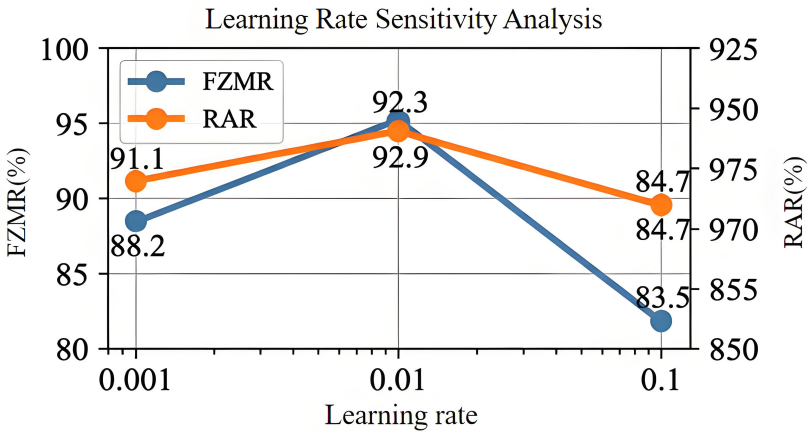


Figure 5. Sensitivity of HCVEA to learning rate, number of layers, and batch size. Both FZMR (left y-axis) and RAR (right y-axis) are plotted to illustrate hyperparameter effects across basic, ablation, and full configurations.

7. Discussion

7.1. Practical Deployment and Robustness

HCVEA addresses the theory–practice divide of generative design tools by emphasizing efficiency, adaptability, and robustness. Parallel A3C training reduces iterations by 32.1%, enabling faster convergence, while multi-scenario validation shows that residential layouts achieve FZMR 90.7% and RAR 92.3%, and office/commercial layouts maintain SUR > 83% and constraint compliance >85%. Custom user constraints, such as accessible design and room orientation preferences, reach 87.2% compliance, and layouts are generated in 140 ms per instance. These results reflect the underlying framework: LSFS ensures latent stability, CTC-Attention preserves spatial coherence, and A3C refines functional rationality, explaining why HCVEA consistently performs well across diverse scenarios. The combination of high performance, adaptability, and constraint compliance demonstrates HCVEA's practical applicability for both experimental and professional design use.

7.2. Implications for Architects and AI

HCVEA has key implications for architectural practice and AI-driven design. For architects, it functions as a decision-support tool that generates layouts, meeting architectural norms while exploring alternative spatial configurations. By quantifying metrics such as FZMR, RAR, and SUR, it provides actionable feedback, accelerates iterative design, and supports complex scenarios like multi-room or structurally constrained layouts, bridging generative experimentation and real-world applications.

From an AI and architectural perspective, CTC-Attention and A3C form a unified framework: CTC-Attention captures spatial patterns for diversity and coherence, while A3C enforces architectural rules for functional rationality and efficient training. This coarse-to-fine design combines creativity with normativity, embedding architectural knowledge and occupant-centered criteria into generative models to produce accurate, practical, and interpretable layouts.

HCVEA uses A3C over methods like PPO because layout refinement is a discrete, incremental task. A3C's parallel exploration and online updates leverage CTC-Attention features, reducing training iterations by 32.1% (Section 7.1) while achieving high refinement accuracy (RAR = 92.3%), showing that the task-specific algorithm design outperforms simply adopting the latest methods.

8. Conclusions

HCVEA leverages an improved Conditional Variational Autoencoder architecture combined with reinforcement learning techniques to generate personalized residential layouts. The proposed model demonstrates improved layout accuracy, flexibility, diversity, and real-time verification capability. Key experimental findings are summarized as follows:

(1) The model achieves an FZMR of 90.7% and an RAR of 92.3%, averaged over three independent runs with low standard deviations (<1.2%), indicating stable and accurate layout generation. This stability and accuracy are further enhanced by the gradient stabilization term, which ensures that small perturbations in the latent space lead to smooth, spatially coherent, and functionally rational layout modifications.
(2) HCVEA maintains a high constraint compliance rate of 88.6%, showing that generated layouts respect spatial and functional rules within the tested scenarios.
(3) Performance is consistent across diverse room configurations and input variations, demonstrating robustness to different layout specifications within the dataset.
(4) Ablation experiments show performance drops when key modules are removed, confirming the importance of the Conditional VAE, CTC-Attention decoder, and A3C. A3C is preferred over methods like PPO due to its parallel exploration and lightweight integration with spatial decoding, enabling efficient and functionally rational layout refinement.
(5) Multi-scenario validation, including residential, office, and commercial layouts, indicates that HCVEA can reliably generate layouts across different types of spaces under the tested conditions.

Beyond performance metrics, HCVEA impacts design, occupants, and AI-driven generative systems. By combining layout generation with real-time verification of metrics and constraints, it supports early-stage decisions, streamlines iterative refinement, and ensures layouts meet functional, environmental, and user-specific requirements. Embedding occupant preferences and accessibility constraints produces personalized, inclusive spaces while engaging users in verification. From an AI perspective, HCVEA demonstrates that probabilistic generative modeling with reinforcement learning and explicit verification can create controllable, occupant-centered systems, balancing creativity and usability.

Limitations remain. Experimental validation relies solely on the FloorNet dataset (2000 plans), restricting generalizability. Cross-dataset performance and regional design styles remain untested. Adaptability to complex scenarios, such as multi-story buildings or specialized functional requirements, is unexplored. Additionally, reinforcement learning increases computational cost, limiting large-scale deployment.

Future work will enhance reliability and applicability by (1) expanding training data with public datasets (e.g., RPLAN, LSUN Layout) and real projects; (2) performing cross-dataset and cross-regional validation; (3) increasing independent runs with statistical testing; (4) extending to multi-story and mixed-use scenarios; and (5) improving computational efficiency. Incorporating real-time user feedback and additional design criteria such as sustainability, energy efficiency, and daylighting will further strengthen occupant-centered relevance.

Author Contributions: Conceptualization, H.H., Z.L. and F.X.; methodology, H.H., Z.L. and F.X.; software, H.H., Z.L. and F.X.; validation, H.H. and Z.L.; formal analysis, H.H. and F.X.; resources, Z.L. and F.X.; data curation, Z.L. and F.X.; writing—original draft preparation, H.H., Z.L. and F.X.; writing—review and editing, H.H., Z.L. and F.X.; visualization, H.H.; supervision, Z.L. and F.X.; project administration, F.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The dataset used in this study is publicly available and can be accessed at the following link: https://github.com/art-programmer/FloorNet (accessed on 10 May 2025).

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

Appendix A. RAR Calculation Rules

This appendix outlines the principles for calculating RAR (Room Arrangement Rationality), forming the quantitative basis for the Functionality Metrics. The rules cover spatial adjacency, functional zoning, and structural feasibility, enabling automated evaluation of layout coherence, zoning continuity, and structural soundness. They ensure Functionality Metrics align with architectural standards and provide interpretable guidance for automatic verification of layout rationality.

Table A1. Category A: spatial adjacency rules.

ID	Rule	Logic
R1	LR–DR adjacency	Share a wall
R2	K–DR adjacency	Share a wall
R3	MB Ensuite	MB next to MBa
R4	Public/private separation	No direct door connection
R5	Service Core proximity	Wet areas near plumbing

Table A2. Category B: functional zoning rules.

ID	Rule	Logic
R6	Zone continuity	Rooms in same zone contiguous
R7	Circulation efficiency	Clear path from entrance to all zones
R8	Natural lighting	Habitable rooms have exterior wall
R9	Dimension compliance	MB $\geq$ 12 m^2, K $\geq$ 6 m^2

Table A3. Category C: structural feasibility rules.

ID	Rule	Logic
R10	Wall alignment	Partition walls align with structural grid
R11	No overlaps	Rooms/structures do not overlap
R12	Accessibility	Corridors $\geq$ 90 cm, doors $\geq$ 80 cm

References

1. De, T.S. Improving Conditional VAE with approximation using Normalizing Flows. *arXiv* **2025**, arXiv:2511.08946. [CrossRef]
2. Hu, Y.; Sun, Y.; Wen, X.; Wang, S.; Bao, J. Shopfloor layout generation method based on large language models. *Int. J. Comput. Integr. Manuf.* **2025**, *39*, 284–302. [CrossRef]
3. Jiang, Z.; Sun, S.; Zhu, J.; Lou, J.-G.; Zhang, D. Coarse-to-Fine Generative Modeling for Graphic Layouts. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 1096–1103. [CrossRef]
4. Lu, J.; Lei, L.; Huang, J.; Yang, F.; Shang, L.; Zeng, X. Automatic Op-Amp Generation From Specification to Layout. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **2023**, *42*, 4378–4390. [CrossRef]
5. Nauata, N.; Chang, K.-H.; Cheng, C.-Y.; Mori, G.; Furukawa, Y. House-GAN: Relational Generative Adversarial Networks for Graph-constrained House Layout Generation. *arXiv* **2020**, arXiv:2003.06988.
6. Komorska, I.; Puchalski, A. Condition Monitoring Using a Latent Space of Variational Autoencoder Trained Only on a Healthy Machine. *Sensors* **2024**, *24*, 6825. [CrossRef]
7. Kwon, Y.; Yoo, J.; Choi, Y.S.; Lee, H.; Kim, Y. Efficient learning of non-autoregressive graph variational autoencoders for molecular graph generation. *J. Cheminform.* **2019**, *11*, 70. [CrossRef]
8. Cao, Y.; Liu, C.; Ma, Y.; Zhang, X.; Wang, L.; Li, Z. Self-refined variational transformer for image-conditioned layout generation. *Int. J. Mach. Learn. Cybern.* **2025**, *16*, 1607–1624. [CrossRef]
9. Ye, Z.; Liu, S.; Tian, Z.; Chen, Y.; Zheng, L.; Chen, J. Graph-RWGAN: A Method for Generating House Layouts Based on Multi-Relation Graph Attention Mechanism. *Buildings* **2025**, *15*, 3623. [CrossRef]
10. Tang, H.; Shao, L.; Sebe, N.; Van Gool, L. Graph Transformer GANs With Graph Masked Modeling for Architectural Layout Generation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 4298–4313. [CrossRef] [PubMed]
11. Yenew, A.B.; Assefa, B.G. From algorithms to architecture: Computational methods for house floorplan generation. *SN Comput. Sci.* **2024**, *5*, 589. [CrossRef]
12. Li, K.; Ye, W. A generative adversarial active learning method for mechanical layout generation. *Neural Comput. Appl.* **2023**, *35*, 19315–19335. [CrossRef]
13. Cheng, W.; Shan, Y. Learning layout generation for virtual worlds. *Comput. Vis. Media* **2024**, *10*, 577–592. [CrossRef]
14. Chen, L.; Jing, Q.; Zhou, Y.; Li, Z.; Shi, L.; Sun, L. Element-conditioned GAN for graphic layout generation. *Neurocomputing* **2024**, *591*, 127730. [CrossRef]
15. Zhao, B.; Yin, W.; Meng, L.; Zhang, L.; Wang, X.; Li, M.; Huang, X. Layout2image: Image Generation from Layout. *Int. J. Comput. Vis.* **2020**, *128*, 2418–2435. [CrossRef]
16. Ayçam, İ.; Akalp, S.; Görgülü, L.S. The application of courtyard and settlement layouts of the traditional Diyarbakır houses to contemporary houses: A case study on the analysis of energy performance. *Energies* **2020**, *13*, 587. [CrossRef]
17. Zakraoui, J.; Saleh, M.; Al-Maadeed, S.; Jaam, J.M. Improving text-to-image generation with object layout guidance. *Multimed. Tools Appl.* **2021**, *80*, 27423–27443. [CrossRef]
18. Qiu, Z.; Liu, J.; Xia, Y.; Qi, H.; Liu, P. Text semantics to controllable design: A residential layout generation method based on stable diffusion model. *Dev. Built Environ.* **2025**, *23*, 100691. [CrossRef]
19. Zeng, P.; Gao, W.; Li, J.; Yin, J.; Chen, J.; Lu, S. Automated residential layout generation and editing using natural language and images. *Autom. Constr.* **2025**, *174*, 106133. [CrossRef]
20. Kam, A.; Tanaka, Y.; Brosque, C.; Roedel, H.; Lau, T.; Fischer, M. Comparison of dusty robotics and traditional layout methods. *Constr. Robot.* **2025**, *9*, 23. [CrossRef]
21. Wu, Z.; Liu, W.; Cheng, J.C.P.; Wang, Z.; Kwok, H.H.L.; Huang, C.; Hou, F. Agile and creative: A sustainability-oriented generative framework for residential site layout design. *Energy Build.* **2025**, *334*, 115499. [CrossRef]
22. Hassan, M.U.; Alaliyat, S.; Hameed, I.A. Image generation models from scene graphs and layouts: A comparative analysis. *J. King Saud Univ.-Comput. Inf. Sci.* **2023**, *35*, 101543. [CrossRef]
23. Zeng, P.; Yin, J.; Huang, Y.; Li, M.; Zhang, W.; Chen, F. AI-based generation and optimization of energy-efficient residential layouts controlled by contour and room number. *Build. Simul.* **2025**, *18*, 2777–2805. [CrossRef]
24. Mak, H.W.L.; Han, R.; Yin, H.H.F. Application of Variational AutoEncoder (VAE) Model and Image Processing Approaches in Game Design. *Sensors* **2023**, *23*, 3457. [CrossRef]

25. Sasazawa, Y.; Sogawa, Y. Layout Generation Agents with Large Language Models. *arXiv* **2024**, arXiv:2405.08037. [CrossRef]
26. Baucum, M.; Khojandi, A.; Vasudevan, R. Improving Deep Reinforcement Learning With Transitional Variational Autoencoders: A Healthcare Application. *IEEE J. Biomed. Health Inform.* **2021**, *25*, 2273–2280. [CrossRef]
27. Wang, X.; Liu, H. Data supplement for a soft sensor using a new generative model based on a variational autoencoder and Wasserstein GAN. *J. Process Control* **2020**, *85*, 91–99. [CrossRef]
28. Liu, H.; Wu, J.; Jiang, J.; Qi, M.; Ren, B. Sequence-based Person Attribute Recognition with Joint CTC-Attention Model. *arXiv* **2018**, arXiv:1811.08115.
29. Shen, H.; Zhang, K.; Hong, M.; Chen, T. Towards Understanding Asynchronous Advantage Actor-Critic: Convergence and Linear Speedup. *IEEE Trans. Signal Process.* **2023**, *71*, 2579–2594. [CrossRef]
30. Liu, C.; Wu, J.; Furukawa, Y. FloorNet: A Unified Framework for Floorplan Reconstruction from 3D Scans. *arXiv* **2018**, arXiv:1804.00090. [CrossRef]
31. Gupta, K.D.; Kamal, M.; Rifat, R.H.; Haque, M.A.; George, R. Physical Rule-Guided Convolutional Neural Network. *arXiv* **2024**, arXiv:2409.02081 [CrossRef]
32. Gupta, K.; Lazarow, J.; Achille, A.; Davis, L.S.; Mahadevan, V.; Shrivastava, A. LayoutTransformer: Layout Generation and Completion With Self-Attention. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 1004–1014.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.