Graph-Based Autoregressive Machine Learning for High Performance Residential Layout Generation

Shafiul Islam

A thesis
submitted in partial fulfillment of the
requirements for the degree of

Master of Science in Architecture

University of Washington

2026

Committee:
Tomás Méndez Echenagucia
Mehlika Inanici
Mohammad Tabatabaei Manesh

Program Authorized to Offer Degree:
Architecture

University of Washington

Abstract

Graph-Based Autoregressive Machine Learning for High Performance Residential Layout Generation

Shafiul Islam

Chair of the Supervisory Committee:

Tomás Méndez Echenagucia

Department of Architecture

The design of residential layouts extends beyond the distribution of space; it also requires the careful coordination of spatial organization and environmental performance. However, most machine-learning-based floor-plan generation methods treat performance as a post-generation evaluation criterion, rather than as an input condition that guides the generation process itself. This thesis develops a prediction-based generative framework that conditions residential layout generation on environmental-performance targets during the generation process.

The proposed framework operates on structured geometric data rather than images. It first predicts room-area shares and then assigns rooms to spatial zones informed by environmental performance metrics using a conditional graph-based autoregressive model. A separate connectivity model predicts room adjacencies, while a schematic synthesis stage assembles candidate layouts from the predicted areas, zones, and connections. The generated layouts are evaluated as a population, and non-dominated alternatives are selected along a Pareto front, preserving competing performance objectives as explicit design trade-offs rather than collapsing them into a single optimized outcome.

This study uses two physically coupled environmental contexts, Quality View and acoustic performance, as representative performance conditions. Because both are affected by spatial arrangement and the window-to-wall ratio, two study-specific environmental performance indicators—the Quality View Coverage Index (QVCI) and the Acoustic Quality Index (AQI)—were developed and incorporated into the machine-learning training process. By demonstrating that these coupled performance conditions can guide layout generation, the framework establishes a basis for extending the same approach to additional environmental contexts in future work.

The performance-conditioned spatial zoning model achieved approximately (86%) accuracy in predicting room-zone assignments, with a cross-entropy loss of (0.20). More importantly, it demonstrated strong target controllability, with target prediction accuracy reaching approximately (94%) and a root-mean-square error of

(0.06) between requested and achieved performance values. Sweeping the conditioning targets produced large and correctly directed changes in the achieved metrics, including directional gaps of (+0.71) and (+0.54) for the two environmental performance axes and (+0.66) for the zone-spread target. The slider-responsiveness results further show that the model can steer both environmental performance and the spatial distribution of rooms across the plan.

These results demonstrate that the proposed pipeline can generate diverse and plausible residential layouts that respond to user-defined performance targets. Rather than producing a layout first and evaluating it afterward, the framework generates candidate layouts already shaped by performance trade-offs. Environmental performance is therefore treated not as a post-generation check, but as a condition that guides generation itself.

Acknowledgements

I would like to express my sincere gratitude to my advisor and committee chair, Professor Tomás Méndez Echenagucia, for his guidance, patience, and insight throughout this research. His expertise in environmental performance and computational design shaped this work at every stage.

I am also grateful to my committee members, Professor Mehlika Inanici and Mohammad Tabatabaei Manesh, for their thoughtful feedback and for the time they devoted to reviewing this thesis.

I thank the Department of Architecture at the University of Washington for providing the academic environment and resources that made this work possible.

I would like to give special thanks to Md Shariful Alam Anik for his constant support and encouragement throughout this journey.

Finally, I thank my family, especially Esrat Jahan, and my friends for their unwavering support and encouragement throughout my graduate studies.

Contents

List of Figures

List of Tables

List of Abbreviations and Acronyms

AI	Artificial Intelligence
AQI	Acoustic Quality Index
BIM	Building Information Modeling
dB	Decibel
GAN	Generative Adversarial Network
GAT	Graph Attention Network
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
ISO	International Organization for Standardization
JSON	JavaScript Object Notation
KL	Kullback–Leibler (divergence)
LLM	Large Language Model
MLP	Multilayer Perceptron
QV	Quality View
QVCI	Quality View Coverage Index
ReLU	Rectified Linear Unit
RMSE	Root Mean Square Error
SPL	Sound Pressure Level
STC	Sound Transmission Class
TL	Transmission Loss
VAE	Variational Autoencoder
WHO	World Health Organization
WWR	Window-to-Wall Ratio

Nomenclature

General

$\mathcal{G}$	Room adjacency graph of a floor plan
N	Number of rooms in a plan
A_i	Floor area of room i
$\|$	Vector concatenation
$\odot$	Elementwise product
$\mathbf{1}\{\cdot\}$	Indicator function
$\mathbf{W}, \mathbf{b}$	Learned weight matrix and bias vector
$\mathcal{L}$	Loss function (subscripted by pipeline stage)

Area distribution

A_{total}	Total floor area of the plan
y_k	Target area fraction of room type k
K	Number of room types
$\mathbf{h}_1, \mathbf{h}_2, \mathbf{h}_3$	Hidden representations of the area network
$\mathbf{p}$	Predicted area-fraction vector (logistic-normal sample)
μ, σ	Mean and standard deviation of the latent Gaussian
ϵ	Standard normal noise sample
S	Number of Monte Carlo samples in the loss

Performance metrics

S_z	Set of exposed sides of zone z
$c(d)$	View-quality weight of the context in direction d
$v(z)$	View weight of zone z
w_i	Window-to-wall ratio of room i
q_i	Quality-view indicator of room i
τ_{QV}	Quality-view threshold (0.15)
L_d	Outdoor sound pressure level on façade direction d
$r_d^{\mathrm{w}}, r_d^{\mathrm{d}}, r_d^{\mathrm{s}}$	Window, door, and solid-wall area ratios of façade d
τ_d	Sound transmission coefficient of façade d
TL_d	Transmission loss of façade d
S_d	Area of façade d
A_{sab}	Sabine absorption area of the room
$L^{\mathrm{in}}(z)$	Indoor sound pressure level in zone z
θ_{t_i}	Acoustic threshold of room type t_i
$L_{\min}$	Lower reference sound pressure level (10 dB)
a_i	Acoustic score of room i
$\mathcal{H}$	Set of acoustically sensitive room types (bedroom, living, kitchen)
n_z	Number of rooms assigned to zone z
$\mathrm{Conc}, \mathrm{Spread}$	Zone concentration and its complement

Zone assignment model

$\mathbf{x}_i, \tilde{\mathbf{x}}_i$	Raw and site-augmented feature vector of room i
$\mathbf{v}$	Vector of view zone weights (9 zones)
$\hat{\mathbf{a}}$	Vector of acoustic zone weights (9 zones)
e_{ij}, α_{ij}	Unnormalized and normalized attention coefficients
$\mathbf{h}_i, \mathbf{h}'_i$	Node embedding before and after a graph attention layer
M	Number of attention heads
$\pi(t)$	Room visited at decoding step t
$z_t, z_{<t}, \mathbf{z}$	Zone at step t, partial sequence, and full assignment
$\mathbf{s}_t$	Recurrent decoder state at step t
$\mathbf{g}_t$	Decoder input at step t
ρ_t	Fraction of rooms remaining at step t
$u_{t,z}$	Usage count of zone z before step t
$\mathbf{c}, \mathbf{c}_t$	Conditioning (budget) vector; its step-t form
$\mathbf{o}$	Achieved performance triple $(\mathrm{QVCI}, \mathrm{AQI}, \mathrm{Spread})$
$\mathbf{t}_{\text{target}}$	Sampled conditioning target
η	Gaussian target-perturbation noise
$w_b, m_{b,t}$	Arrangement-type weight and validity mask in the loss
$\mathbf{u}, \beta$	Latent variable and KL weight of the conditional VAE

Connectivity

$\mathbf{A}$	Room adjacency matrix
b_{ij}	Binary edge indicator between rooms i and j
$\mathcal{E}_{\text{order}}$	Lower-triangular edge visiting order
$\mathbf{r}_{ij}, \mathbf{q}_{ij}$	Edge input and recurrent edge state
p_{ij}	Predicted edge probability

Evaluation

ℓ, ℓ'	Candidate layouts
$\succ$	Pareto dominance relation
$\mathcal{F}$	Pareto front of non-dominated candidates
$\Delta_{\mathrm{QVCI}}, \Delta_{\mathrm{AQI}}$	Conditioning response gaps between high and low targets

Chapter 1

Introduction

1.1 Background

With advances in design computation, the generation of architectural layouts has evolved from rule-based and generative algorithms toward contemporary machine learning approaches. Machine-learning-based architectural layout generation has become an active topic in academic research. Alongside this research, commercial design tools are increasingly offering automated floor plan generation capabilities. However, most of these tools operate primarily as space allocators: they distribute spaces according to function, area efficiency, and connectivity requirements.

A floor plan, however, is more than an allocation of space. In practice, architects position rooms within a site by also accounting for the environmental requirements of each space—daylight, quality view, acoustic comfort, thermal comfort, ventilation, and other site-related conditions. Most existing methods assess environmental performance only after the layout has been generated, rather than using environmental performance criteria to guide the generation process from the outset. Comparatively little research has examined how environmental performance metrics can be used as input conditions that actively guide the layout generation process.

1.2 Research Gap

Although environmental performance is central to architectural design, existing research rarely treats it as a driver of the layout generation process. A review of the current literature reveals three recurring gaps in performance-aware layout generation:

- Environmental performance metrics are omitted entirely during layout generation.
- Environmental performance is calculated or simulated only after a layout has been generated, rather than informing it.
- Floor plans are treated as image data, which is computationally expensive and carries no explicit representation of environmental performance.

From a workflow perspective, environmental performance simulation remains a complex and repetitive part of the design process. In both generative design workflows and traditional trial-and-error methods, environmental simulation typically occurs after the layout has been generated. Consequently, when the simulated performance proves unsatisfactory, the designer must modify or regenerate the layout and run the simulation again—a cycle that is frequently repeated. In multi-objective optimisation, these repeated simulation steps become computationally expensive and time-consuming.

1.3 The Machine Learning Opportunity

This simulation-intensive cycle can potentially be reduced if layouts are generated directly from quantitative environmental-performance criteria. Within building-performance research, surrogate modelling has been widely used to approximate simulation-based performance outcomes at substantially lower computational cost, reducing the need for repeated expensive simulations (Westermann and Evins 2019). Prediction-based layout generation therefore offers a promising means of addressing the workflow gap described above.

1.4 Research Objective

Building on the gaps and opportunities identified above, this study pursues a single overarching aim. The objective of this study is to propose a scalable, prediction-based layout generation framework that generates architectural layouts based on environmental performance metrics.

To achieve this objective, the study is guided by the following specific goals. First, it aims to integrate a generative machine-learning architecture capable of producing new architectural layout arrangements. Second, it seeks to develop a framework that can incorporate an extendable number of environmental performance metrics, allowing future expansion beyond the metrics tested in this study. Third, it aims to establish a scalable layout generation approach that can accommodate variations in site boundary complexity and differences in the number of rooms within a plan. Finally, the study aims to demonstrate the potential of

performance-informed layout generation as an alternative to conventional workflows in which environmental performance is evaluated only after the layout has already been generated.

In this context, scalability refers to the ability of the proposed framework to extend across environmental metrics, datasets, site geometries, building scales, and future machine-learning methods. The framework is intended to accommodate additional performance criteria beyond the representative metrics tested in this study, such as daylighting, thermal comfort, or energy demand. It is also designed to remain applicable as larger floor-plan datasets become available and as site boundaries vary in size and geometric complexity. More broadly, while this thesis focuses on residential layouts, the approach is developed as a generalisable framework that can support future adaptation to other building types and improved learning architectures.

1.5 Scope

This study adopts residential units within single apartments as its test case. To bound the scope of environmental performance, two metrics are examined: quality view and acoustic performance. These serve as representative environmental performance variables; if both can be successfully integrated into the proposed framework, it is hypothesised that additional environmental metrics could be incorporated in future work.

Two considerations motivate the selection of these metrics. First, both depend on windows and exhibit a reciprocal relationship with the window-to-wall ratio (WWR): increasing window area improves quality view but degrades acoustic performance by raising the indoor sound pressure level (SPL). This inherent trade-off makes the two metrics well suited to testing a multi-objective framework. Second, both are strongly dependent on the position of rooms within the site, since quality view and acoustic performance each mediate the relationship between indoor rooms and outdoor conditions. Together, these metrics provide a focused test case for evaluating whether the proposed framework can handle performance conditions that influence room placement and spatial organisation.

1.6 Research Questions

To address the gaps outlined above, this research is guided by the following two research questions.

1. How can architectural layouts be generated using a prediction-based machine learning framework that is conditioned on environmental performance metrics during—rather than after—the generation process?

2. How can layouts be generated from structured geometric data rather than image-based representations, so that the generation process remains computationally lightweight and explicitly aware of environmental performance?

1.7 Contribution and Significance

The proposed framework is a prediction-based scheme that generates residential layouts from environmental, spatial, and performance-related features. Machine learning is used to (i) assign rooms to spatial zones based on user-defined environmental performance targets, and (ii) learn the relationships between layout configuration and connectivity. The framework then generates candidate layout arrangements that combine these predictions to approximate the target performance, after which higher-performing options can be identified by plotting performance across the generated layouts.

The principal contributions are:

- A prediction-based layout generation framework.
- A method for integrating Quality View and acoustic performance into the layout generation process.
- A demonstrative test case using residential apartment units.
- A scalable framework designed to accommodate additional environmental metrics in future work.
- A workflow that reduces dependence on repeated post-generation simulation.
- An interactive web and desktop application that presents multiple layout alternatives to support early-stage project initiation and design exploration.

In terms of significance, this research helps architects make informed decisions during the early phase of design. Collectively, the framework aims to shift the paradigm from automatic space allocation toward performance-informed layout generation—making the generation process more performance-aware, strengthening the link between machine learning and architectural decision-making, and reducing reliance on simulation-based trial and error.

1.8 Limitations

Although the proposed framework is designed to be scalable and extendable, the current implementation has several methodological and technical limitations.

First, the framework is demonstrated using selected environmental performance metrics rather than the full range of environmental factors that influence architectural layout design. In this study, Quality View and acoustic performance are used to test the proposed approach.

Second, the quality of the generated layouts depends on the quality, consistency, and diversity of the training data. If the dataset contains limited variations in spatial organisation, boundary conditions, room

relationships, or geometric configurations, the model may reproduce those limitations in the generated outputs.

Third, the environmental performance values used in the generation process are based on study-specific indicators developed to represent Quality View and acoustic performance within the machine-learning workflow. Rather than using full simulation outputs directly, this thesis adopts custom performance metrics that can be consistently encoded, learned, and compared within the generative model. These indicators support early-stage design exploration, but they are not intended to replace detailed environmental simulation or post-design validation.

1.9 Thesis Organisation

This thesis is organised into five chapters. Chapter 1 introduces the research problem, objective, scope, contributions, and limitations. Chapter 2 reviews prior work on computational layout generation, machine-learning-based floor-plan generation, and performance-informed design. Chapter 3 presents the proposed methodology, including data preparation, feature definition, model development, layout synthesis, and evaluation procedures. Chapter 4 presents and discusses the results, including model performance, target controllability, generated layout evaluation, and performance trade-offs. Chapter 5 concludes the thesis and outlines directions for future research.

Chapter 2

Literature Review

2.1 Introduction

This chapter reviews several overlapping bodies of literature and integrates them into a coherent, unified argument. It first surveys computational and learning-based layout generation and shows that most methods remain centred on image representations, with little attention to performance-driven design. Commercial tools and performance-based design workflows are then examined to show that the same limitation extends beyond academic research: environmental performance is typically evaluated after a layout has been generated, rather than used to guide generation itself. Together, these academic, industrial, and performance-based design threads define the research gap addressed by this thesis. The chapter then turns to the methodological opportunity. It reviews the machine-learning foundations that support performance-conditioned generation. These include learned performance surrogates, graph neural networks, autoregressive generation, conditional learning, preference-based learning, link prediction, and a cross-domain analogy to molecular machine learning. The chapter closes by examining available datasets for floor-plan learning and the evaluation and selection of generated designs. It then presents a positioning statement that maps the identified research gaps to the specific mechanisms adopted in this study.

2.2 Computational Layout Generation: An Overview

Computational approaches to architectural layout generation can be grouped into three broad families: rule-based, optimisation-based, and learning-based methods (Liu et al. 2025). Rule-based methods generate layouts from explicit design rules, such as shape grammars or procedural constraints. Optimisation-based methods search through possible layouts and evaluate them against predefined objective functions. Learning-

based methods, by contrast, train models on datasets of existing plans in order to learn spatial patterns and generate new layouts. This thesis belongs to the learning-based family. This distinction is important because optimisation-based methods often require repeated evaluations during the search process. When those evaluations depend on environmental simulation, the workflow can reproduce the same computational bottleneck that this thesis seeks to reduce.

2.3 Learning-Based Floor-Plan Generation

Learning-based floor-plan generation has advanced rapidly since 2020, progressing through graph-conditioned generative adversarial networks (GANs), image-to-image translation, graph- and vector-native generation, diffusion models, and most recently large-language-model and tokenised approaches. Early graph-constrained GANs such as House-GAN (Nauata et al. 2020) and its iterative successor House-GAN++ (Nauata et al. 2021) take a room-adjacency graph as input and emit per-room raster masks. Graph2Plan (Hu et al. 2020) similarly conditions on a user-supplied layout graph but predicts room boxes that are refined into a floor plan, marking an early move toward structured outputs. Image-to-image pipelines such as ArchiGAN (Chaillou 2019) translate a rasterised parcel into a rasterised plan through a stack of Pix2Pix models. A subsequent line of work abandons the pixel grid altogether: WallPlan (Sun et al. 2022) synthesises plans by generating wall graphs directly, and HouseDiffusion (Shabani et al. 2023) denoises vector room coordinates with a diffusion model. Raster diffusion methods generate or in-paint rasterised plans under various conditions (Zeng et al. 2024; Shim et al. 2024; Yenew et al. 2024). The most recent work introduces graph-attention GANs (Ye et al. 2025), two-stage LLM-plus-diffusion pipelines (Zong et al. 2024), and tokenised multimodal LLMs (Qin et al. 2026).

Two limitations emerge from this review. First, although recent methods increasingly adopt vector or graph-native representations (Hu et al. 2020; Sun et al. 2022; Shabani et al. 2023), graph reasoning is used to encode spatial structure — adjacency, walls, and room geometry — rather than environmental behaviour; the graph carries topology, not performance. Second, environmental performance is only weakly incorporated: none of the reviewed methods uses quantified daylight, thermal, view, or acoustic performance targets as generative conditions. These limitations define the gap to which this study responds.

To maintain consistency across the reviewed studies, the environmental-performance column in Table 2.1 is evaluated using an ordinal scale based on two criteria: whether environmental performance is quantified, and whether it can be specified as a condition that steers generation. “No” indicates that the method does not include any environmental quantity. “Very limited” and “weak” indicate that an environmental signal is present only implicitly, such as a sunlight cue embedded in conditioning imagery, without a quantified

metric or user control over its value. "Partial" indicates that a measurable, performance-adjacent quantity is included in the pipeline, such as connectivity structure or a guidance term in the diffusion process, but is not provided as an explicit user-defined performance target. "Conceptual" indicates that environmental performance is discussed as a motivation or framing, while the model itself does not operationalise any performance quantity. Under this scale, none of the reviewed methods reaches the level targeted in this thesis: quantified environmental-performance metrics supplied as explicit conditions during generation.

Table 2.1: Review of learning-based floor-plan generation methods.

Method	**Year**	**Method family**	**Addressed Env performance**	**Image-based**	**Representation**
House-GAN (Nauata et al. 2020)	2020	Graph GAN	No	Yes — output	Raster
House-GAN++ (Nauata et al. 2021)	2021	Constraint GAN (iterative)	Very limited	Yes — output	Raster
Graph2Plan (Hu et al. 2020)	2020	Graph-conditioned (boxes + refinement)	No	Partial (raster refinement)	Hybrid
WallPlan (Sun et al. 2022)	2022	Wall-graph generation	No	No	Vector graph
HouseDiffusion (Shabani et al. 2023)	2023	Diffusion (vector coordinates)	No	No	Vector
ArchiGAN (Chaillou 2019)	2019	GAN (Pix2Pix stack of 3)	No	Yes — in & out	Raster
Multi-cond. Diffusion (Zeng et al. 2024)	2024	Diffusion	Weak (sunlight hint)	Yes — output	Raster
FloorDiffusion (Shim et al. 2024)	2024	Diffusion (PEFT + in-painting)	No	Yes — in & out	Raster
HouseGanDi (Yenew et al. 2024)	2024	GAN + Diffusion hybrid	No	Yes — output	Hybrid
Graph-RWGAN (Ye et al. 2025)	2025	Graph-attention GAN	Partial (connectivity)	Yes — output	Raster
HouseTune (Zong et al. 2024)	2025	LLM + Diffusion (two-stage)	Partial	Partial (diffusion)	Hybrid
HouseMind (Qin et al. 2026)	2026	Multimodal LLM (tokenised)	Conceptual	Partial (tokenised)	Hybrid

2.4 Commercial and Industry Tools

Alongside academic research, a growing number of commercial tools now generate or assist with floor-plan layouts. Based on publicly available product documentation, these tools can be grouped into three broad categories: real-estate feasibility and yield optimisers, such as TestFit and Hypar (TestFit, Inc. 2026; Hypar, Inc. 2026); AI-assisted generative layout tools, such as Maket, ARK, Finch, Snaptrude, Conix.AI, SWAPP, Spacio, and Qbiq (Maket 2026; ARK 2026; Finch 3D 2026; Snaptrude 2026; Conix.AI 2026; SWAPP 2026; Spacio 2026; Qbiq 2026); and environmental-analysis-integrated platforms, most notably Autodesk Forma (Autodesk 2025). The first two groups operate primarily as space allocators, distributing area according to

function, code compliance, connectivity, and development yield, but without conditioning layout generation on environmental performance. Forma is the instructive exception: it supports early-stage analysis of sun hours, daylight, wind, noise, and embodied carbon, and recent versions extend this workflow to interior layout exploration (Autodesk 2025). Critically, however, Forma analyses the performance of a massing or layout already created by the user and returns the results as design guidance; it does not generate layouts conditioned on achieving a specified performance target. This distinction between performance-as-feedback and performance-as-condition is central to the present study.

Table 2.2: Commercial floor-plan tools.

Tool	**Category**	**Layout Generation**	**Environmental performance**
TestFit	Feasibility / yield optimiser	Yes	None documented
Hypar	Feasibility / generative	Yes	Limited
Finch	AI generative	Yes	None documented
Maket	AI generative	Yes	None documented
Snaptrude	BIM / modelling assist	Modelling-assist	None documented
ARK	AI generative	Yes	None documented
Qbiq	AI generative (test fits)	Yes	None documented
Conix.AI	AI generative	Yes	None documented
SWAPP	AI generative	Yes	None documented
Spacio	AI generative	Yes	None documented
Autodesk Forma	Site / early-stage + analysis	Yes (massing)	Yes, but post-hoc analysis (sun, daylight, wind, noise, carbon) — not a generation condition

2.5 Performance-Based and Performance-Driven Design

A separate strand of research treats environmental performance as a central design concern. This work is rooted in simulation-based design, where early-stage building simulation is used to understand the performance consequences of design decisions (Østergård et al. 2016). Performance-driven design extends this logic by coupling simulation with search or optimisation. In these workflows, design variables are tested against quantified objectives, such as heating, cooling, daylighting, or competing multi-objective performance criteria (Evins 2013; Shi and Yang 2013; Nguyen et al. 2014). Generative design systems apply a similar logic to spatial layout by producing large populations of candidate plans and evaluating them against measurable goals (Bao et al. 2013; Nagy et al. 2017). More recently, commercial platforms have also begun to integrate environmental analysis into early-stage modelling workflows, as discussed in Section 2.4.

However, two practical constraints remain. First, physical simulation can be computationally expensive, especially when many design alternatives must be evaluated. This has motivated the use of surrogate models that approximate simulation outputs at a lower computational cost (Wortmann et al. 2015; Westermann and Evins 2019), as well as simplified evaluation metrics designed for generative workflows (Natanian and Wortmann 2021). Second, and more importantly for this thesis, performance usually enters the workflow

only after a candidate layout has already been produced. A designer or algorithm generates a layout, evaluates its performance, and then revises or regenerates the design if the result is unsatisfactory. Even when optimisation or surrogate modelling is used, performance often acts as a downstream filter rather than an upstream condition for generating the layout itself.

The present study reframes this relationship. Rather than generating layouts first and evaluating them afterward, it supplies quantified performance targets as generative conditions. In this formulation, performance intent helps shape the layout at the moment of generation. This connects the performance-driven design tradition reviewed here with the learning-based generation methods discussed in Section 2.3, while addressing a key limitation shared by both: the lack of a direct connection between performance targets and layout generation.

2.6 Research Gap

Bringing these threads together, learning-based floor-plan generation has matured in both representation and method (Section 2.3), but it continues to treat environmental performance as absent or post-hoc. Commercial tools remain largely space allocators, and even the strongest exception analyses performance rather than generating from it (Sections 2.4–2.5). At the same time, machine learning can predict environmental performance cheaply, and the modelling tools needed to act on that prediction—graph attention, autoregressive generation, conditional and preference-based learning, and link prediction—are mature, with a clear cross-domain precedent in molecular discovery; these foundations are reviewed in Section 2.7. What remains missing is a framework that quantifies environmental performance, supplies it as a condition during generation, and operates on lightweight attributed-graph data rather than images. This is the gap addressed by the present study. The remainder of this chapter reviews the foundations that make filling it feasible, and Section 2.12 maps each element of the gap to the mechanism adopted in response.

2.7 From Performance Prediction to Conditioned Generation

Machine learning has become an important tool for predicting building-performance metrics, including daylight, energy, and related environmental quantities, at a fraction of the computational cost of physical simulation. Recent work further demonstrates that a learned daylight surrogate can be paired with a generative model (Hu et al. 2024). The remaining gap is not simply prediction, but the integration of prediction into generation: using performance estimates to condition layout generation rather than evaluating performance only after a layout has been produced.

2.7.1 Sequence Models and Autoregressive Generation

Autoregressive models generate an output one element at a time, with each step conditioned on what has been produced so far. The paradigm originates in sequence modelling — sequence-to-sequence learning (Sutskever et al. 2014), the gated recurrent unit (Cho et al. 2014), and the Transformer (**vaswani2017attention**) — and extends to structured generation in images (Oord et al. 2016), graphs (You et al. 2018b; Liao et al. 2019), and layouts (Gupta et al. 2021). Its defining property is that generation is decomposed into an ordered series of conditional decisions, each one shaped by the partial structure already in place.

This conditional, one-decision-at-a-time structure has a natural parallel in architectural design. A layout is rarely fixed in a single simultaneous act; instead, a designer commits spatial decisions incrementally, each choice constrained by those already made and by the existing conditions of the site. The plan emerges as an ordered sequence of dependent decisions rather than all at once. Autoregressive generation mirrors this reasoning directly, which makes it a fitting paradigm for the layout problem and motivates its use in this study.

Within floor-plan generation specifically, sequential and iterative generation has already been adopted: RPLAN generates rooms one after another in a learned order (Wu et al. 2019), and House-GAN++ refines a layout through successive conditional passes (Nauata et al. 2021).

2.7.2 Cross-Domain Analogy: Molecular Machine Learning

A close structural parallel exists in computational chemistry, where molecules are represented as graphs and where both molecular properties and molecular structures are modelled using methods closely related to those relevant here. Property prediction commonly relies on message-passing graph neural networks (Gilmer et al. 2017; Yang et al. 2019), in which each atom repeatedly updates its state from its bonded neighbours. This message-passing principle is itself a form of local, context-dependent reasoning: the representation of any one element is shaped by the elements adjacent to it, much as a spatial decision in a layout depends on its immediate surroundings rather than on the plan in isolation. It also provides a precedent for approximating expensive evaluations with learned predictors.

Molecular generation extends the same graph- and sequence-based logic, using models such as the variational JT-VAE (Jin et al. 2018), the reinforcement-learning-based GCPN (You et al. 2018a), the autoregressive flow model GraphAF (Shi et al. 2020), and the transformer-decoder MolGPT (Bagal et al. 2022). The autoregressive members of this family build a structure one element at a time under learned constraints — the same conditional, incremental paradigm that characterises architectural decision-making, in which each addition is made in the context of what is already present.

The parallel is informative but bounded. The analogy has clear limits. Molecular validity is governed by strict chemical constraints and can be tested through physical assays, whereas floor-plan validity is more flexible and context-dependent. In this study, performance values are represented through study-specific analytical indicators rather than fixed physical ground truth. This distinction is stated explicitly to avoid overstating the transfer from molecular machine learning to architectural layout generation.

2.7.3 Representations of Floor Plans

The internal representation of a plan is not a neutral technical choice; it shapes what a model can generate, condition, and evaluate. Across learning-based layout generation, four representational families recur: raster representations, boundary or footprint conditioning, topology graphs, and attributed graphs. Raster methods treat the plan as a pixel grid; boundary-conditioned methods rely on a fixed site or apartment outline; topology graphs represent rooms as nodes and adjacencies as edges; and attributed graphs enrich this structure with quantitative geometric and semantic features. Although some learning-based generators reason over graphs internally, their final outputs are still commonly rendered as pixels, with vector geometry recovered only through post-processing. Thus, the key limitation is not that existing methods are purely image-based, but that graph reasoning rarely becomes the primary generative representation.

The present work addresses this limitation by conditioning generation on an attributed graph that carries environmental, geometric, and performance-related information, including target values for environmental performance. Because it does not require a fixed boundary image, the framework is positioned closer to an early-stage design process, where spatial organisation and performance intent can be explored before geometric resolution is fixed.

2.7.4 Graph Neural Networks and Graph Attention

Graph neural networks learn representations of data structured as nodes and edges. The foundational graph convolutional network (Kipf and Welling 2017) aggregates neighbourhood information through a localised spectral approximation, and GraphSAGE (Hamilton et al. 2017) extends this to an inductive setting. The graph attention network (Veličković et al. 2018) introduces masked self-attention so that a node can weight its neighbours unequally. Because rooms in a plan form a natural graph with heterogeneous relationships, attention-based graph encoders are well suited to representing per-room features together with inter-room structure, weighting each neighbour according to its relevance rather than treating all connections equally.

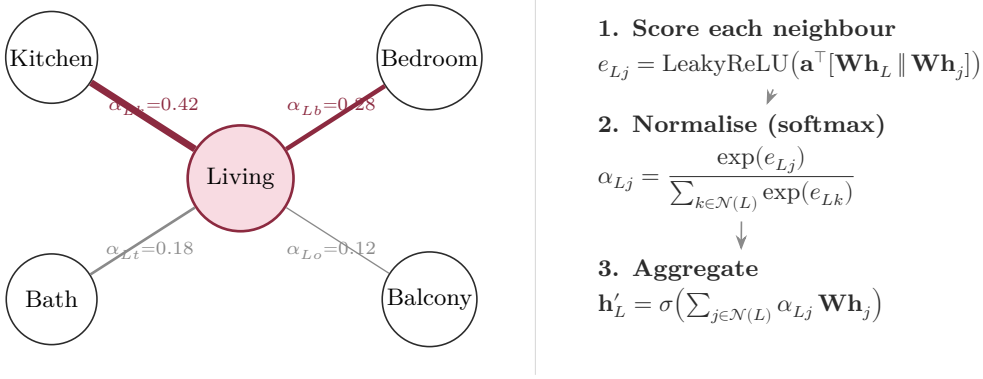


Figure 2.1: The graph-attention mechanism. Each neighbour of a node is scored, the scores are normalised into attention weights α_{Lj}, and the node's updated representation is a weighted sum of its neighbours — so adjacent rooms contribute unequally according to learned relevance. Mechanism after Veličković et al. (2018).

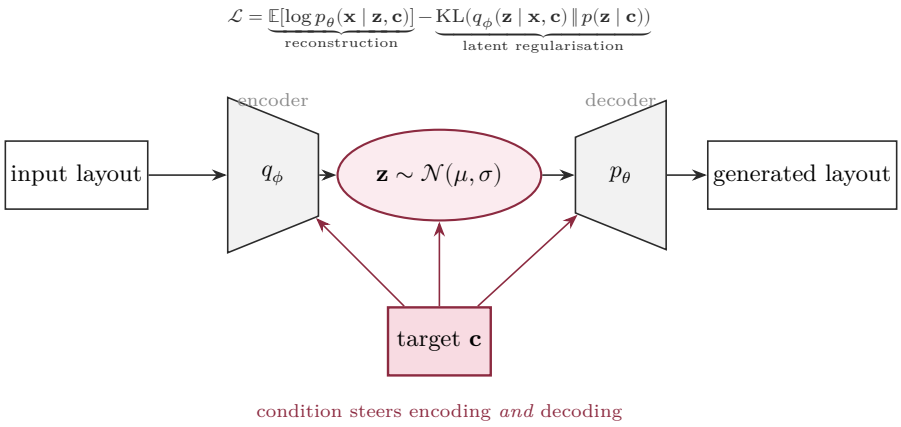


Figure 2.2: The conditional variational autoencoder. An encoder maps an input layout to a latent distribution and a decoder reconstructs from a sample; the condition $\mathbf{c}$ is supplied to both, so generation is steered toward a specified target. The objective combines a reconstruction term with a KL term that regularises the latent space. Formulation after Sohn et al. (2015).

2.7.5 Variational Autoencoders and Conditional Generation

The variational autoencoder (Kingma and Welling 2014) learns a continuous latent space from which new samples can be drawn, and the conditional variational autoencoder (Sohn et al. 2015) conditions that generation on an observation such as a label or attribute. Conditioning generation on a target attribute is what allows a model to produce diverse outputs that nonetheless satisfy a specified requirement, making conditional generation a natural fit for problems in which a layout must respond to externally given performance goals.

2.7.6 Ranking and Discriminative Scoring Models

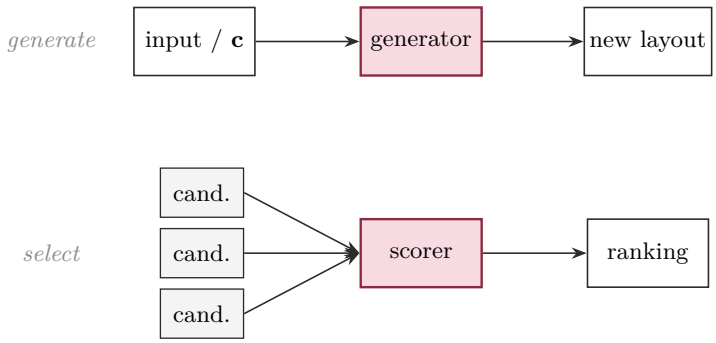


Figure 2.3: Generative and discriminative approaches are complementary. A generative model proposes a new layout from an input or condition, whereas a discriminative ranking model scores and orders a set of existing candidates, selecting among them rather than producing new ones.

An alternative to generating layouts is to *score* candidate arrangements and rank them. Learning-to-rank methods such as RankNet (Burges et al. 2005), surveyed by Liu (2009), learn ordering relations from labelled pairs. Rather than producing a layout directly, such discriminative approaches learn to evaluate and order existing candidates, offering a complementary route to generation: where generative models propose arrangements, ranking models select among them.

2.7.7 Link Prediction and Architectural Adjacency Graphs

Predicting which rooms should connect is a graph link-prediction problem. Node-based approaches such as the variational graph auto-encoder (Kipf and Welling 2016) and subgraph-based approaches such as SEAL (Zhang and Chen 2018) learn to infer missing edges from graph structure and node features. In architecture these edges correspond to adjacency and access relationships — the bubble-diagram tradition. Link prediction is therefore the natural formulation for inferring the connectivity that a layout requires, generating these relationships from structure rather than assuming them as a given input.

2.8 Datasets for Floor-Plan Learning

Most learning-based methods reviewed here train on a small number of public datasets, notably RPLAN (Wu et al. 2019) and, for language-conditioned work, Tell2Design (Leng et al. 2023). These datasets provide geometry and room types but not environmental attributes or contrasting good/bad arrangements. The present study instead builds on the more recent ResPlan dataset (Abouagour and Garyfallidis 2025), a large-scale vector-graph corpus of residential plans with explicit window, balcony, and connectivity annotations. ResPlan was selected because its room-area and window-to-wall-ratio distributions are comparatively consistent across the corpus, and area and window-to-wall ratio are the two attributes on which the performance metrics of this study most directly depend.

2.9 Contrastive and Preference Learning

A further body of work concerns learning from comparisons between examples rather than from absolute labels. Contrastive representation learning (Chen et al. 2020) trains models to separate positive from negative examples, and preference learning formalises ordering from paired comparisons through the Bradley–Terry model (Bradley and Terry 1952), which underpins modern preference-based training (Christiano et al. 2017). These foundations justify constructing data from contrasting arrangements of the same plan, for example a high-performing arrangement paired against a low-performing one, so that a model can be exposed to the full performance range rather than to high-quality examples alone.

2.10 Evaluating and Comparing Generative Models

Generative models require a basis for comparison. In the wider generative-modelling literature, models are assessed on validity, diversity, novelty, and fidelity to the conditioning signal, and distributional metrics such as the Fréchet Inception Distance (Heusel et al. 2017) and, in the molecular domain, benchmark suites such as GuacaMol (Brown et al. 2019) and MOSES (Polykovskiy et al. 2020) operationalise these properties. Together these establish the vocabulary — validity, diversity, and fidelity to a target — in which any performance-conditioned generator must ultimately be evaluated.

2.11 Multi-Objective Optimisation and Design-Space Exploration

Environmental objectives frequently compete, so that no single layout is optimal on all of them at once. Multi-objective methods address this through the concept of Pareto optimality, in which a set of non-

dominated solutions represents the best available trade-offs rather than a single best answer. Generating a population of candidate layouts and identifying those on the Pareto front is an instance of performance-based design-space exploration, the paradigm on which a principled treatment of competing objectives rests.

Table 2.3: Mapping of identified gaps to the mechanisms adopted in this study.

Gap	**Mechanism adopted**
Environmental performance rarely quantified	Physically grounded performance metrics computed for every plan
Performance evaluated only after generation	Performance targets supplied as conditions during generation
Image-centred representations	Attributed room graph as the primary generative representation

2.12 Positioning of This Study

The gap articulated in Section 2.6 has three elements: environmental performance is rarely quantified, it is never supplied as a condition during generation, and plan representations remain image-centred. Each is answered by a specific mechanism reviewed in this chapter. Performance is quantified through physically grounded view and acoustic metrics computed for every plan. Performance enters generation as a condition rather than a post-hoc filter: quantified targets are supplied to the generator as part of its input, drawing on conditional generation (Section 2.7.5), with rooms assigned to zones sequentially by an autoregressive decoder (Section 2.7.1). The plan is represented as an attributed graph rather than an image: a graph-attention encoder (Section 2.7.4) carries per-room environmental and geometric features, so graph reasoning is the primary generative representation rather than an internal intermediate. Table 2.3 summarises this mapping; the methodology that follows develops each mechanism in turn.

Chapter 3

Methodology

3.1 Overview of the Framework

The central methodological aim of this work is to use quantified environmental performance as a condition for layout generation, rather than as a measure applied after a layout has been produced. In conventional performance-based design, a layout is first generated or arranged and then evaluated, often requiring repeated simulation and revision. The framework developed in this thesis reverses this sequence. Target performance values are provided as inputs, and the generative models are trained to produce layouts that respond to those targets.

Generating a complete residential layout in a single step is difficult because the output couples geometry, topology, and performance simultaneously. The framework therefore decomposes the problem into four sequential prediction stages, each individually conditionable and individually trainable:

1. **Area distribution** (Section 3.2) — predicting how the total floor area is partitioned among the rooms;
2. **Zone assignment** (Section 3.3) — placing each room in an appropriate zone of the site under environmental targets, here into one of the nine spatial zones of a 3×3 grid;
3. **Connectivity** (Section 3.4) — predicting the room-adjacency graph, including the connection to the exterior;
4. **Schematic synthesis** (Section 3.5) — assembling the predicted area, aspect ratio, window-to-wall ratio (WWR), zone, and connectivity into geometric rectangles, and selecting high-performing candidates from a two-objective Pareto front.

The stages are trained once, offline, and then chained at inference to generate a population of candidate layouts. The two environmental performance metrics that the pipeline targets — the Quality View Coverage Index (QVCI) and the Acoustic Quality Index (AQI) — together with an auxiliary structural control on how the rooms are spread across the site, are introduced where they are first used, in Section 3.3.2.

Figure 3.1 shows the prediction-based generation scheme that runs at inference, and Figure 3.2 shows the full methodology workflow, distinguishing the offline data-preparation and training phases from the online generation and evaluation phases.

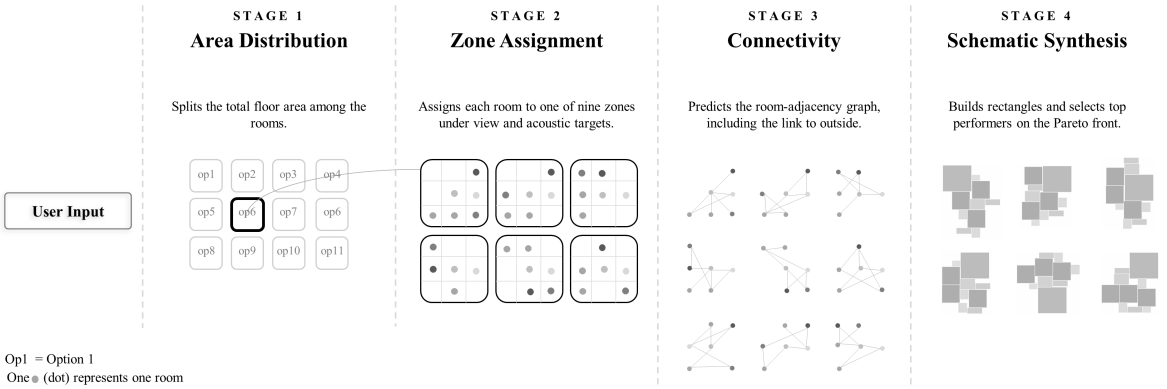


Figure 3.1: The prediction-based generation scheme used at inference. Program inputs flow through area, zone, and connectivity prediction into schematic layouts, which are filtered by a two-objective (QVCI–AQI) Pareto front.

3.2 Stage 1 — Area Distribution Prediction

The first stage answers a single question: given the program of a dwelling — how many rooms of each type it contains and the total interior area — how should that area be divided among the rooms? Its output seeds the geometry of every later stage, so it must be both plausible and diverse.

3.2.1 Dataset Preparation for Area Distribution

The area model is trained on (program, area-split) pairs extracted from the corpus of residential plans used throughout this chapter; the cleaning and feature extraction applied to that corpus are detailed in Section 3.3.1. For each plan the program is summarised by the vector of room-type counts $\mathbf{n}$ and the total interior area A_{total}, and the supervision target is the vector of per-room-type *area shares* obtained by normalising the measured room areas to sum to one. Because room counts are heavy-tailed, the counts are compressed by a logarithmic transform, $\mathbf{x} = \log(1+\mathbf{n})$, before being passed to the model. The training target $\mathbf{y}$ therefore lies on the probability simplex $\Delta^{K-1} = \{\mathbf{y} : y_k \geq 0,\ \sum_k y_k = 1\}$, where K is the room-type vocabulary size.

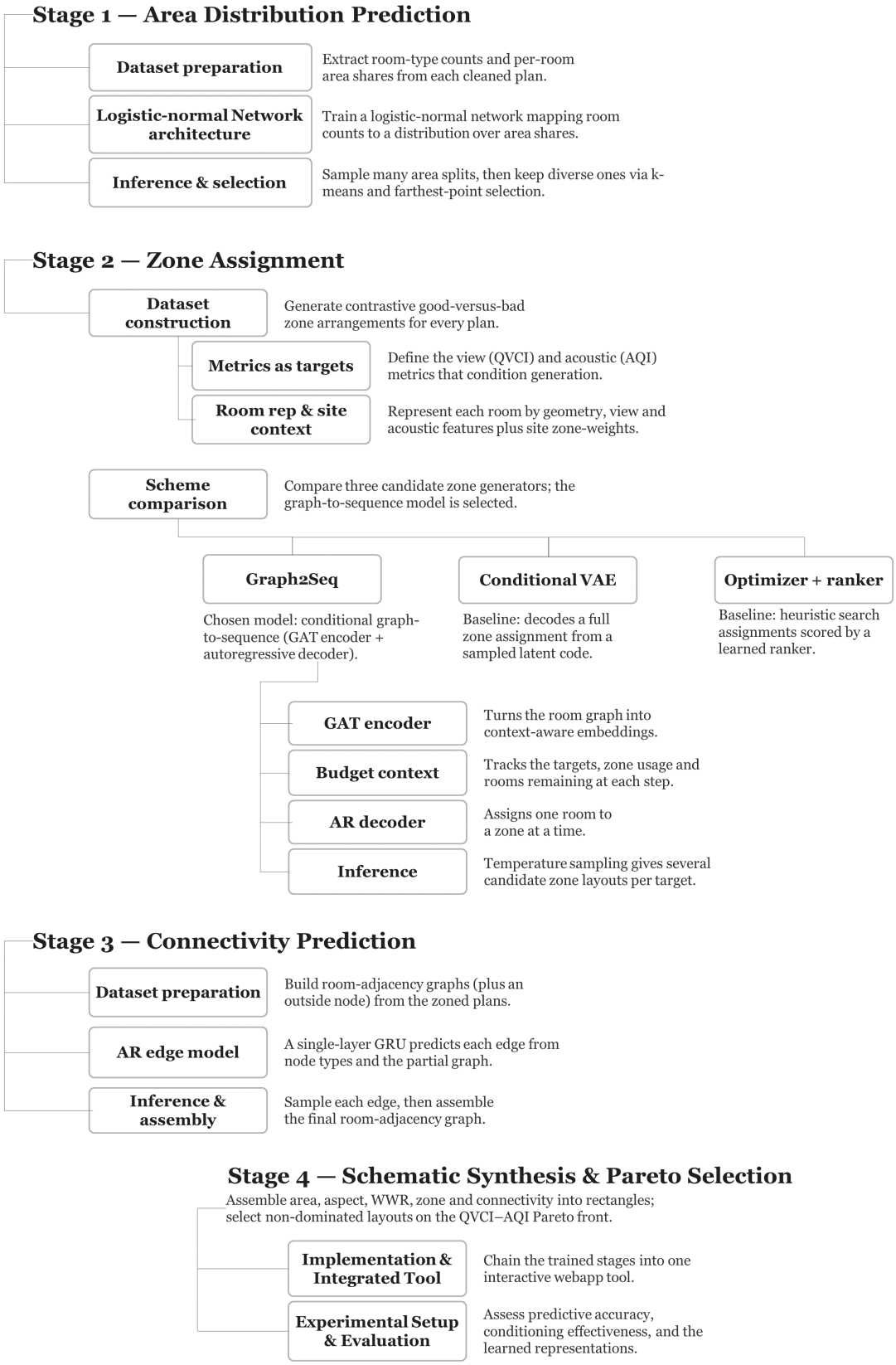


Figure 3.2: The full methodology workflow. The data-preparation and model-training phases are performed once, offline; the generation phase produces candidate layouts; and the evaluation phase assesses them.

3.2.2 Logistic-Normal Formulation

Predicting a point on the simplex, rather than K unconstrained areas, builds the conservation constraint directly into the model: predicted shares are guaranteed non-negative and sum to one, and absolute areas follow by rescaling,

$$A_k = y_k \, A_{\text{total}}. \tag{3.1}$$

A useful distribution over the simplex must capture both the typical split and the correlations between room types (a larger living room usually means a smaller third bedroom). The model therefore predicts a *logistic-normal* distribution: a Gaussian in unconstrained logit space mapped onto the simplex by the softmax. An encoder maps the input to a latent vector and two heads produce a mean μ and a log-standard-deviation $\log\sigma$; a sample of shares is drawn by perturbing the mean with Gaussian noise in logit space and applying the softmax,

$$\mathbf{p} = \text{softmax}(\mu + \sigma \odot \epsilon), \qquad \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), \qquad \sigma = \exp(\log\sigma), \tag{3.2}$$

where $\odot$ is the elementwise product. The mean μ captures the most likely split, while σ controls how much it may vary — the source of diversity exploited in Section 3.2.5.

3.2.3 Network Architecture

The encoder is a compact multilayer perceptron. The compressed counts pass through two fully connected layers of width 128 with ReLU activations and dropout ($p = 0.1$), are projected to a 16-dimensional latent vector, and are read out by the two heads:

$$\begin{aligned}
\mathbf{h}_1 &= \text{dropout}\big(\,\text{ReLU}(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1)\big),\\
\mathbf{h}_2 &= \text{dropout}\big(\,\text{ReLU}(\mathbf{W}_2\mathbf{h}_1 + \mathbf{b}_2)\big),\\
\mathbf{h}_3 &= \mathbf{W}_3\mathbf{h}_2 + \mathbf{b}_3, \qquad \mu = \mathbf{W}_\mu\mathbf{h}_3 + \mathbf{b}_\mu, \qquad \log\sigma = \mathbf{W}_\sigma\mathbf{h}_3 + \mathbf{b}_\sigma,
\end{aligned} \tag{3.3}$$

with $\log\sigma$ clamped to $[-5, 1]$ for numerical stability. The 16-dimensional bottleneck acts as a low-dimensional embedding of the program, in which programs with similar area logic sit close together. Figure 3.3 shows the network as a whole, and Figure 3.4 the weighted-sum-and-activation unit from which every layer is built.

3.2.4 Training Objective

Because the prediction is a distribution rather than a point, the model is trained to make samples from that distribution match the observed share vector. The loss is a Monte-Carlo estimate of the expected

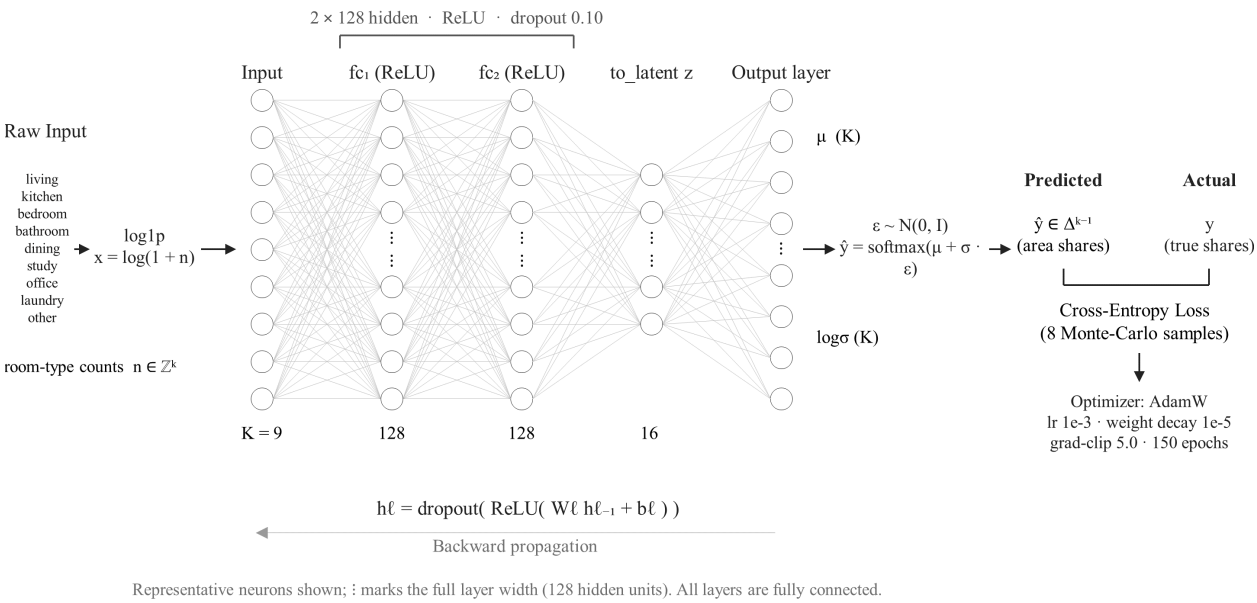


Figure 3.3: The area model as a layered network. Compressed room-type counts pass through two hidden layers to a low-dimensional latent vector, from which the mean and log-standard-deviation heads parameterise a logistic-normal distribution over area shares.

cross-entropy between the target shares $\mathbf{y}$ and the sampled predicted shares: for each example, $S = 8$ noise vectors are drawn, pushed through Equation (3.2), and scored against $\mathbf{y}$,

$$\mathcal{L}_{\text{area}} = \frac{1}{S}\sum_{s=1}^{S}\left(-\sum_{k=1}^{K} y_k \log p_k^{(s)}\right), \qquad \mathbf{p}^{(s)} = \text{softmax}(\mu + \sigma \odot \epsilon^{(s)}). \tag{3.4}$$

Averaging over the samples rewards a predicted distribution whose typical draw is close to the truth, so the network places probability mass where the real splits lie while still expressing uncertainty through σ. The model is trained for 150 epochs with the AdamW optimiser (learning rate 10^{-3}, weight decay 10^{-5}), a batch size of 256, gradient-norm clipping at 5.0, and a 90/10 train/validation split.

3.2.5 Inference and Representative Selection

At inference the program fixes μ and σ, and drawing many noise vectors through Equation (3.2) yields a population of plausible splits, each rescaled to absolute areas by Equation (3.1). Because most samples are near-duplicates, a small set of mutually distinct representatives is selected before passing them downstream, using k-means clustering of the share vectors (the centroids of the main modes) and farthest-point sampling (the most dissimilar splits). The result is a handful of genuinely different starting points rather than

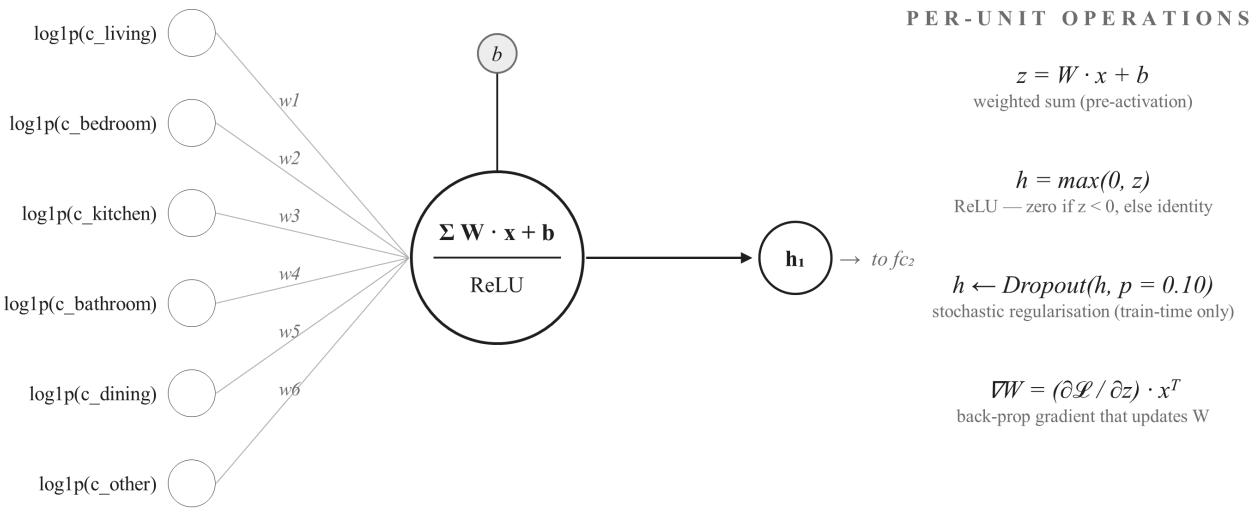


Figure 3.4: Anatomy of a single fully-connected unit: a weighted sum of its inputs plus a bias, followed by a ReLU activation (and dropout at training time). Every hidden unit of the area network follows this pattern; the linear heads use the same weighted sum without the activation.

Figure 3.5: Sample Floor Plan from Dataset.

redundant variants of one answer.

3.3 Stage 2 — Zone Assignment

Zone assignment is the core stage of the framework and the one that carries the performance conditioning: given the rooms of a plan and target performance values, it assigns each room to one of the nine zones of a 3×3 grid. This section first describes how the labelled, contrastive dataset is constructed (Section 3.3.1), then defines the metrics that serve as conditioning targets (Section 3.3.2), compares the candidate modelling schemes (Section 3.3.3), and finally details the adopted model component by component (Sections 3.3.4–3.3.11).

3.3.1 Dataset Construction

3.3.1.1 Source plans, cleaning, and window recovery

The corpus is derived from the ResPlan dataset of residential floor plans (Abouagour and Garyfallidis 2025), a large-scale collection of vector-format plans with annotated walls, doors, windows, and balconies, from which 16,734 plans survive cleaning and filtering. Each room is described by a 30-dimensional feature vector. Twenty-one of its entries are numerical: geometry (area, normalised width and height, normalised centroid coordinates, number of exposed facades), view features (the overall WWR and its four per-direction components), acoustic features (the acoustic WWR and its four per-direction components), four derived quantities (area fraction, view potential, indoor SPL, and acoustic score), and a binary balcony-access flag. The remaining nine entries form a one-hot room-type encoding. A large fraction of the raw plans recorded

a WWR of zero for more than half of their rooms because of a broken window-to-room association; this was repaired by widening the window-proximity tolerance, and, for exposed primary rooms that remained at zero, by imputing a WWR sampled from the empirical distribution of the same room type and injecting a corresponding window polygon (about 6.5% of rooms, each flagged as synthetic). This recovery is necessary because the metrics of Section 3.3.2 are undefined or trivially zero for plans whose primary rooms carry no glazing.

3.3.1.2 Context-Driven Zone Decomposition

Before rooms can be assigned, the plan boundary must be divided into spatial zones. Rather than imposing a fixed grid, the decomposition is driven by the environmental context adjacent to each part of the boundary, so that the number and arrangement of zones follow from the site itself.

The principle is that a zone should be homogeneous in the environmental conditions it experiences. Each segment of the boundary faces a particular external context — an orientation, a view condition, a noise source — and wherever that adjacent context changes, a new zone begins. A simple rectangular plan with four uniform sides yields the familiar arrangement of four corner zones, four edge zones, and one interior zone, for nine in total (Figure 3.6, left). The corners are treated as distinct zones because they are exposed to two directions at once, and the interior as a single sheltered zone because it faces none.

By construction, this scheme can scale in two independent ways. First, when a single side of the boundary spans more than one context — for example, part of an edge faces an open view while the remainder faces an adjacent building — that side can be split into multiple zones, one per distinct condition. Second, when the boundary itself is more complex, such as an L-shaped or otherwise concave footprint, each additional corner and edge introduced by the geometry generates its own zone. The elongated plan in Figure 3.6 (centre) introduces additional edge zones along its longer sides, and the L-shaped plan (right) adds further corner and edge zones at its re-entrant junction, reaching fifteen zones. In every case the zones are produced by the same rule; only their count changes.

Although the zones are spatial regions, they form an ordered structure that can be treated as a simple sequence. Traversing the boundary in a fixed order and appending the interior, the zones unwrap into a one-dimensional list (Figure 3.6, bottom), regardless of how many there are or how complex the boundary is. This unwrapping is what allows a plan of any zone count to be processed by the sequential assignment model of Section 3.6: the model operates on a list of zones rather than a grid, and is therefore not tied to a particular zone count by its architecture.

It should be emphasised that the variable-zone cases shown in Figure 3.6 illustrate the generality of the representation rather than the scope of the experiments reported here. In this study, the model is trained and

evaluated exclusively on the nine-zone case corresponding to rectangular plan boundaries, which matches the predominant geometry of the dataset described in (Section 3.6). The decomposition and the sequence-based assignment model are designed so that plans with more zones can be accommodated without architectural change, but training and validation on complex boundaries are left to future work (Section 3.6).

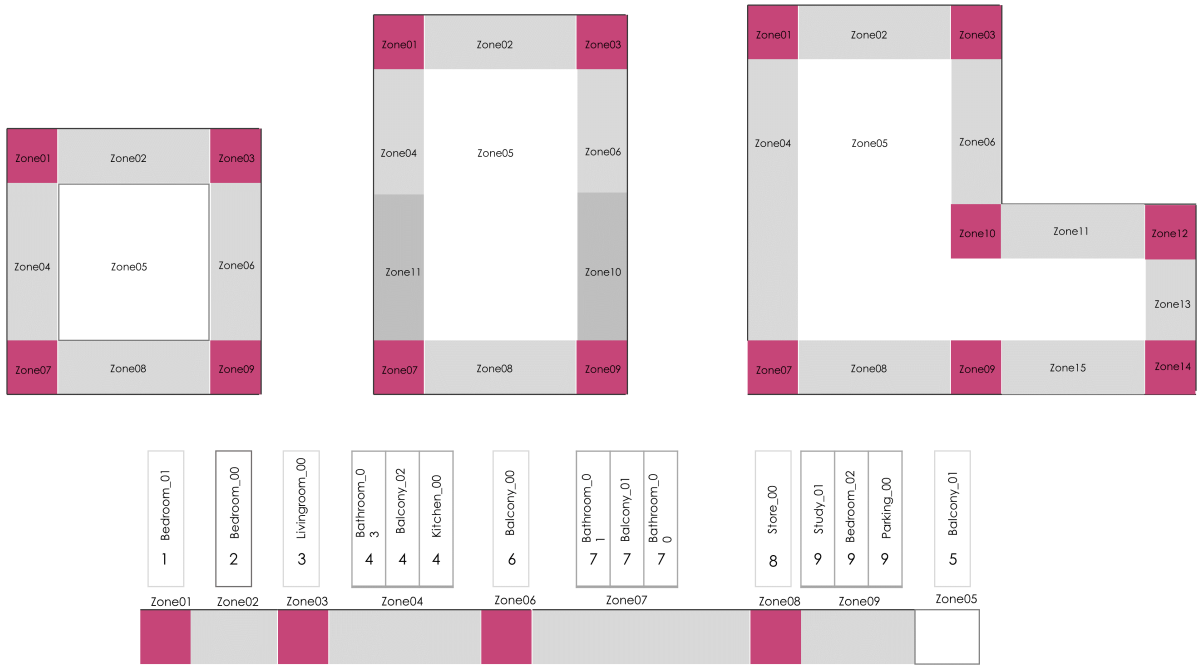


Figure 3.6: Context-driven zone decomposition. Zones are defined by the environmental context adjacent to each boundary segment, so their number follows from the site: a uniform rectangular plan yields nine zones (left), an elongated plan introduces additional edge zones (centre), and a concave, L-shaped boundary adds corner and edge zones at its re-entrant junction, reaching fifteen (right). Traversing the boundary in a fixed order unwraps the zones into an ordered list (bottom). The nine-zone case is used for training and evaluation in this study; the larger cases illustrate the generality of the representation.

3.3.1.3 Contrastive arrangement generation

Because the objective is to teach the model what *good* and *bad* zone assignments look like, each plan is expanded into a family of contrasting arrangements rather than a single label. From every plan, multiple arrangement types are generated — the *real* geometric assignment; small *perturbations*; *greedy-best* and *greedy-worst* assignments under a combined view-and-acoustic objective; *acoustic-greedy* best and worst pairs (isolating the acoustic signal); degenerate *all-in-one* and *all-in-two-zone* assignments; a *type-clustered* assignment; and *constrained-random* and pure *random* assignments. The achieved QVCI, AQI, and spread are computed for each arrangement (Section 3.3.2), so that for the same geometry the model sees arrangements whose performance differs widely.

3.3.1.4 Symmetry augmentation and audit

To reduce directional bias, each floor plan is augmented through mirrored and rotated versions. These transformations follow the eight symmetries of the dihedral group D_4. The view and noise fields are transformed consistently with each plan, so that the augmented versions remain architecturally equivalent. This process improves approximate direction invariance and reduces the north-west exposure bias in the original corpus.

The resulting corpus contains 2,811,312 records. Before model training, an automated audit checks three conditions: whether the performance gap between the best and worst arrangement deciles meets the required minimum, whether direction invariance remains within tolerance, and whether the zone-label distribution remains balanced. If any check fails, the pipeline stops, since regenerating the dataset is cheaper and safer than training on corrupted data (Figure 3.7).

3.3.2 Performance Metrics as Conditioning Targets

Both performance metrics, and the structural spread control, are computed over the 3×3 zoning. The nine zones are indexed $z \in \{1, \dots, 9\}$ in row-major order (NW, N, NE, W, C, E, SW, S, SE); zone 5 is the sheltered interior zone. Each zone is associated with the set of exposed cardinal facades $S_z \subseteq \{\mathrm{N}, \mathrm{S}, \mathrm{E}, \mathrm{W}\}$: corner zones expose two facades, edge zones one, and the interior zone none.

The zone stage is conditioned on three quantities: the two environmental metrics defined in the subsections below — the Quality View Coverage Index and the Acoustic Quality Index — and a purely structural measure of how distributed the rooms are across the grid, the *spread*. Writing n_z for the occupancy of zone z,

$$\mathrm{Conc}(\mathbf{z}) = \frac{\max_z n_z}{N}, \qquad \mathrm{Spread}(\mathbf{z}) = 1 - \mathrm{Conc}(\mathbf{z}), \tag{3.5}$$

so a spread near 1 indicates rooms scattered across many zones and a spread of 0 indicates all rooms collapsed into one. Unlike the two environmental metrics, spread is not an objective of the final selection (Section 3.5); it is a structural control that lets the user request more or less dispersed layouts.

3.3.2.1 Quality View Coverage Index

View quality is assigned to each cardinal direction by a categorical weight $c(\cdot) \in [0, 1]$ (Benedikt 1979; Heschong Mahone Group 2003; Ko et al. 2022): vegetation = 1.0, sky = 0.9, urban = 0.7, balcony = 0.4, no

Figure 3.7: Audit of the zone corpus: coverage of the achieved performance metrics, per-arrangement-type means, and the balance of the zone labels. A corpus that fails any of these checks is regenerated before training.

view = 0.0. A zone's view weight is the mean over the directions it is exposed to,

$$v(z) = \begin{cases} \dfrac{1}{|S_z|} \displaystyle\sum_{d \in S_z} c(d), & S_z \neq \emptyset, \\ 0, & S_z = \emptyset. \end{cases} \tag{3.6}$$

A room realises a quality view only if its glazing is sufficient and its zone faces a worthwhile view; for room i with area A_i, WWR w_i, and zone z_i,

$$q_i = \mathbf{1}\{\, v(z_i)\, w_i \geq \tau_{\mathrm{QV}} \,\}, \qquad \tau_{\mathrm{QV}} = 0.15, \tag{3.7}$$

with $\mathbf{1}\{\cdot\}$ the indicator function. The QVCI is the area-weighted fraction of rooms that realise a quality view,

$$\mathrm{QVCI} = \frac{\sum_{i=1}^{N} A_i\, q_i}{\sum_{i=1}^{N} A_i} \in [0, 1]. \tag{3.8}$$

3.3.2.2 Acoustic Quality Index

Each cardinal direction d carries an outdoor sound pressure level L_d (in dB), drawn from a 62–80 dB urban range. Sound enters through facades, each a composite of window, door, and solid-wall area with fractions $r_d^{\mathrm{w}}, r_d^{\mathrm{d}}, r_d^{\mathrm{s}}$ summing to one; using the sound-transmission-class constants $\mathrm{STC}_{\mathrm{win}}$, $\mathrm{STC}_{\mathrm{door}}$, $\mathrm{STC}_{\mathrm{wall}}$, the composite transmission coefficient and facade transmission loss are

$$\tau_d = r_d^{\mathrm{w}}\, 10^{-\mathrm{STC}_{\mathrm{win}}/10} + r_d^{\mathrm{d}}\, 10^{-\mathrm{STC}_{\mathrm{door}}/10} + r_d^{\mathrm{s}}\, 10^{-\mathrm{STC}_{\mathrm{wall}}/10}, \tag{3.9}$$

$$\mathrm{TL}_d = -10\, \log_{10}(\, \max(\tau_d,\, 10^{-12})). \tag{3.10}$$

With the Sabine absorption area $A_{\mathrm{sab}} = \max(\alpha_{\mathrm{room}}(2A_{\mathrm{px}} + PH), 1)$, the *zone-dependent* indoor level of a room placed in an exposed zone z is

$$L^{\mathrm{in}}(z) = 10\, \log_{10}\left(\sum_{d \in S_z} 10^{(L_d - \mathrm{TL}_d + 10\log_{10}(S_d/A_{\mathrm{sab}}))/10} \right), \tag{3.11}$$

while the interior zone is treated as a solid wall against the mean field, $L^{\mathrm{in}}(z_5) = \overline{L_{\mathrm{out}}} - \mathrm{STC}_{\mathrm{wall}} - 6$, making it the quietest placement. The acoustic index is evaluated over the three primary living spaces — bedroom, living room, and kitchen — as these are the rooms for which a quiet interior is most critical. Each carries a comfort threshold θ_t (World Health Organization Regional Office for Europe 2009; International Organization for Standardization 2024) (bedroom 35 dB; living and kitchen 40 dB), and the per-room acoustic score and

plan-level AQI are

$$a_i = \mathrm{clip}\left(\frac{\theta_{t_i} - L_i^{\mathrm{in}}}{\theta_{t_i} - L_{\min}},\, 0,\, 1\right), \qquad \mathrm{AQI} = \frac{\sum_{i\in\mathcal{H}} A_i\, a_i}{\sum_{i\in\mathcal{H}} A_i}, \tag{3.12}$$

with $L_{\min} = 10\,\mathrm{dB}$ and $\mathcal{H} = \{\mathrm{bedroom}, \mathrm{living}, \mathrm{kitchen}\}$ the set of primary rooms scored for acoustics. For decoder context the per-zone outdoor energy average $\tilde{L}(z)$ is range-normalised to a zone acoustic weight $\hat{a}(z) \in [0, 1]$, which spreads the nine zones evenly across the interval rather than clustering them near 0 and 1. Figure 3.8 shows example view and acoustic zone-weight grids.

Figure 3.8: Example view and acoustic zone-weight grids over the 3×3 zoning. Exposed edge and corner zones carry higher view weight but also admit more outdoor noise, while the sheltered interior zone (zone 5) is the quietest placement.

3.3.2.3 The View–Acoustic Trade-off

Equations (3.8) and (3.12) are coupled through the WWR and zone exposure. A room earns view coverage only when it is well glazed and placed in an exposed zone facing a good view; yet exposed zones admit the most outdoor noise (Equation (3.11)), and additional glazing lowers the transmission loss (Equation (3.10)). Maximising QVCI therefore tends to depress AQI, so the framework seeks not one "best" layout but a set of non-dominated layouts that expose the trade-off (Section 3.5).

3.3.3 Comparison of Candidate Schemes

Three schemes for producing conditioned zone assignments were implemented and compared (Table 3.1) before one was adopted.

3.3.3.1 Conditional Variational Autoencoder

A conditional VAE (Sohn et al. 2015; Kingma and Welling 2014) encodes a known arrangement and its condition into a latent distribution and decodes zone logits, trained on the evidence lower bound

$$\mathcal{L}_{\mathrm{CVAE}} = \mathbb{E}_{q_\phi(\mathbf{u}|\mathbf{z},\mathbf{c})}[\log p_\theta(\mathbf{z} \mid \mathbf{u}, \mathbf{c})] - \beta\,\mathrm{KL}(q_\phi(\mathbf{u} \mid \mathbf{z}, \mathbf{c}) \,\|\, p(\mathbf{u} \mid \mathbf{c})), \tag{3.13}$$

with latent $\mathbf{u}$ and a weight β. At generation the latent is sampled from the prior. This scheme gave the weakest conditioning of the three: sampled assignments did not reliably track the requested targets without careful tuning.

3.3.3.2 Deterministic Optimiser with a Learned Ranker

The second scheme separates feasibility from realism. A deterministic search (greedy placement, swap search, constrained random sampling) enumerates candidates *guaranteed* to satisfy the targets, and a trained ranking network (Burges et al. 2005) scores each for architectural realism. This guarantees target satisfaction but is not generative: its diversity is bounded by the enumerator and the search cost grows with plan size and constraint count.

3.3.3.3 Conditional Graph-to-Sequence Model (Graph2Seq)

The third scheme, and the one adopted, formulates assignment as conditional sequence generation over a graph: a graph attention encoder produces site-aware room embeddings, and an autoregressive decoder emits the zone labels one room at a time. It is chosen because it is *scalable* (one network handles plans of varying size in a single pass), *conditionable* (targets enter the decoder at every step and vary continuously at inference), and *generative* (it learns an explicit factorised distribution from which diverse samples are drawn). Sections 3.3.4–3.3.11 describe it in full.

Table 3.1: Comparison of the three candidate schemes for conditioned zone assignment.

Scheme	**Generative**	**Target control**	**Scales with size**	**Output diversity**
Conditional VAE	Yes	Weak	Yes	Moderate
Optimiser + ranker	No	Exact (by search)	No	Bounded by search
Graph2Seq (GAT + AR) **[adopted]**	Yes	Strong	Yes	High

3.3.4 Room Representation and Site Context

Each room enters the model as the 30-dimensional feature vector of Section 3.3.1. So that every room is aware of the site it sits in, this per-room vector is augmented, before encoding, with the two plan-level zone-weight vectors — the nine view weights $v(z)$ (Equation (3.6)) and the nine acoustic weights $\hat{a}(z)$ — broadcast identically to every room. The encoder therefore receives, per room,

$$\tilde{\mathbf{x}}_i = [\, \mathbf{x}_i \parallel \mathbf{v} \parallel \hat{\mathbf{a}} \,] \in \mathbb{R}^{30+9+9} = \mathbb{R}^{48}, \tag{3.14}$$

where $\parallel$ denotes concatenation. Carrying the site context on every node lets the encoder reason jointly about a room and the quality of the zones it might occupy.

3.3.5 Graph Attention Encoder

The rooms of a plan form the nodes of a fully connected graph: each room may attend to every other room in the same plan, with padding positions masked, so the encoder learns which room–room relationships matter rather than being given a fixed adjacency. A two-layer multi-head graph attention network (Veličković et al. 2018) transforms the augmented node features. For one head, the attention coefficient between rooms i and j and its normalisation over the neighbourhood are

$$e_{ij} = \mathrm{LeakyReLU}(\mathbf{a}^\top [\mathbf{W}\mathbf{h}_i \parallel \mathbf{W}\mathbf{h}_j]), \tag{3.15}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{N} \exp(e_{ik})}, \qquad \mathbf{h}_i^{\mathrm{head}} = \sum_{j=1}^{N} \alpha_{ij}\, \mathbf{W}\mathbf{h}_j, \tag{3.16}$$

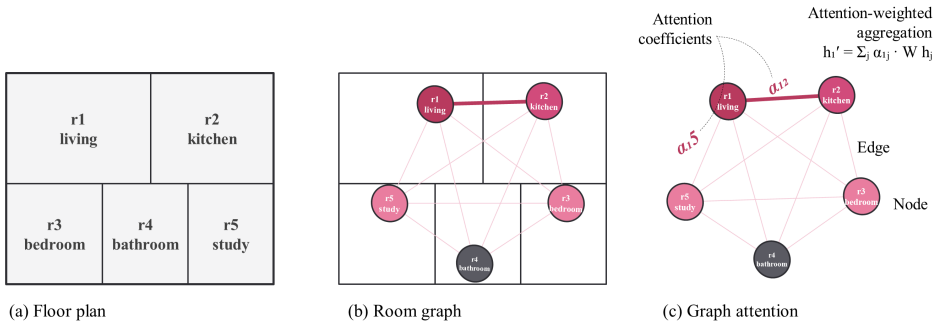
where $\mathbf{W}$ is a learned projection and $\mathbf{a}$ a learned attention vector. With $M = 4$ heads, the updated embedding concatenates the per-head aggregations, with a residual connection, layer normalisation, a ReLU, and dropout:

$$\mathbf{h}_i' = \mathrm{ReLU}\Big(\mathrm{dropout}\,\big(\mathrm{LayerNorm}\,\big(\mathbf{h}_i + \big\|_{m=1}^{M} \mathbf{h}_i^{\mathrm{head}(m)}\big)\big)\Big). \tag{3.17}$$

Two layers are stacked, each at width 128, so the encoder emits one 128-dimensional site-aware embedding per room; two layers let information propagate over two hops, which suffices on the small graphs of residential plans. Figure 3.9 illustrates this attention over an example room graph.

3.3.6 Autoregressive Decoder

The decoder places rooms one at a time, conditioning each placement on all previous ones. Rooms are visited in a fixed order (primary habitable rooms first), and at step t the input for room $\pi(t)$ concatenates



(a) Floor plan (b) Room graph (c) Graph attention

Figure 3.9: From floor plan to attended room graph. Each room becomes a node and the rooms form a fully connected graph; a query room aggregates its neighbours with learned attention coefficients α_{ij} (edge weight), yielding a site-aware embedding.

its encoder embedding, the budget context $\mathbf{c}_t$ (Section 3.3.7), and the one-hot encoding of the zone emitted at the previous step:

$$\mathbf{g}_t = \left[\, \mathbf{h}'_{\pi(t)} \,\|\, \mathbf{c}_t \,\|\, \mathrm{onehot}(z_{t-1}) \,\right] \in \mathbb{R}^{128+31+9} = \mathbb{R}^{168}. \tag{3.18}$$

A two-layer gated recurrent unit (Cho et al. 2014) of hidden width 256 carries the running state, and a small head ($256 \to 128 \to 9$ with a ReLU) produces a distribution over the nine zones:

$$\mathbf{s}_t = \mathrm{GRU}(\mathbf{g}_t, \mathbf{s}_{t-1}), \tag{3.19}$$

$$P(z_t \mid z_{<t}, \mathcal{G}, \mathbf{c}) = \mathrm{softmax}(\mathbf{W}_2\ \mathrm{ReLU}(\mathbf{W}_1 \mathbf{s}_t)). \tag{3.20}$$

The full assignment distribution factorises autoregressively over the rooms, in the manner of sequence and graph generators (Sutskever et al. 2014; You et al. 2018b; Bagal et al. 2022),

$$P(\mathbf{z} \mid \mathcal{G}, \mathbf{c}) = \prod_{t=1}^{N} P(z_t \mid z_{<t}, \mathcal{G}, \mathbf{c}). \tag{3.21}$$

Figure 3.10 shows the encoder–decoder as a whole, and Figure 3.11 the gated recurrent cell at the heart of the decoder.

3.3.7 Budget Context

The budget context $\mathbf{c}_t \in \mathbb{R}^{31}$ is the channel through which the targets and decoding progress reach each step. It concatenates the conditioning target triple $(\mathrm{target}_{\mathrm{QVCI}}, \mathrm{target}_{\mathrm{AQI}}, \mathrm{target}_{\mathrm{spread}})$, the fraction of rooms still to be placed, a running normalised count of rooms already assigned to each zone, and the plan-level view

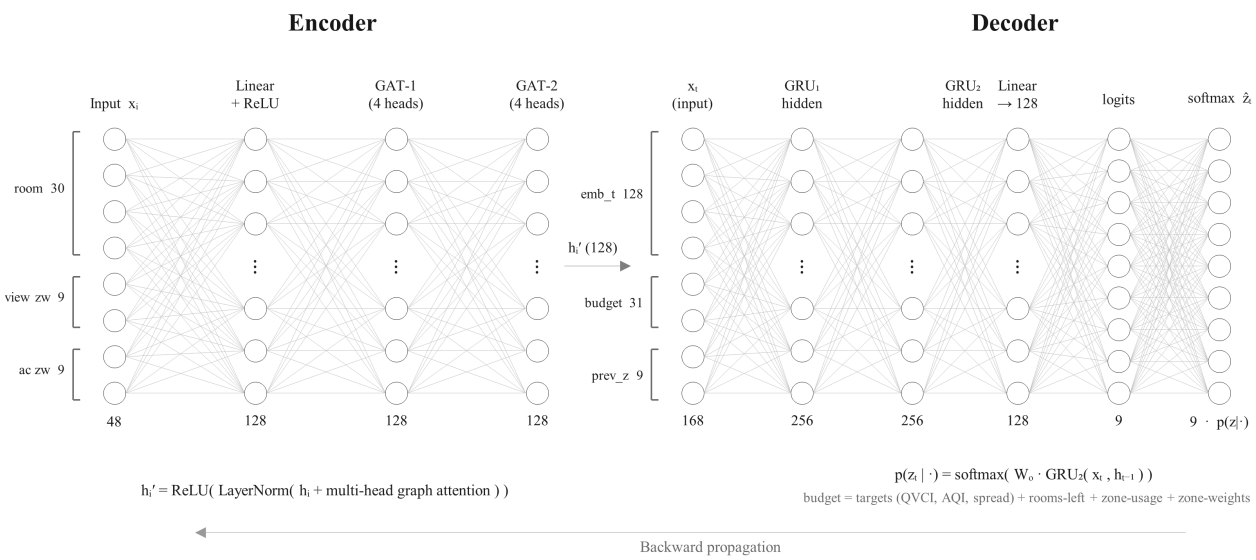
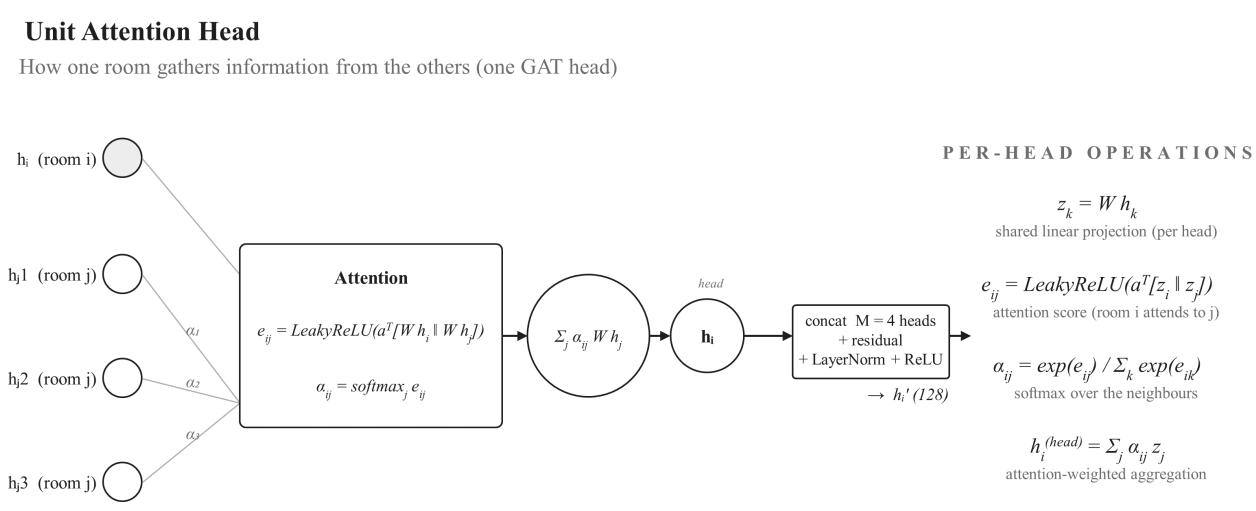


Figure 3.10: The zone model as a layered network. A two-layer graph attention encoder produces a site-aware embedding per room, and a two-layer GRU decoder emits a zone distribution for each room in turn, conditioned at every step on the budget context.



M = 4 heads run in parallel and are concatenated; a residual connection, LayerNorm and ReLU give h_i'. Two such layers are stacked (d = 128). Every room attends to every other room in the plan.

Figure 3.11: Anatomy of a single gated recurrent unit (GRU cell), the recurrent core of the zone decoder. Reset and update gates control how the previous hidden state and the current step input combine into the new state.

and acoustic zone-weight vectors:

$$\rho_t = \frac{N-t}{N}, \qquad u_{t,z} = \frac{1}{t}\sum_{r=1}^{t-1} \mathbf{1}\{z_r = z\}, \qquad \mathbf{c}_t = \big[\, \mathbf{t}_{\text{target}} \,\|\, \rho_t \,\|\, \mathbf{u}_t \,\|\, \mathbf{v} \,\|\, \hat{\mathbf{a}} \,\big]. \tag{3.22}$$

The running zone usage $\mathbf{u}_t$ is a structural progress signal — "how full is each zone so far?" — not a feature derived from the answer; it is the device that lets the decoder respect the requested spread as it proceeds.

3.3.8 Target Conditioning and Leakage Avoidance

The decoder is conditioned on a triple of *target outcomes* — a target QVCI, AQI, and spread — presented as goals rather than as descriptions of the label. This raises a decisive risk: an earlier design conditioned on the outcomes computed directly from the same arrangement used as the label, and because that mapping is invertible, the model could recover the label from its own condition, so conditioning collapsed into memorisation. The present design breaks this with a 70/30 sampling rule applied afresh on every training access. Let $\mathbf{o} = (\text{QVCI}, \text{AQI}, \text{Spread})$ be the achieved outcomes of the labelled arrangement; then

$$\mathbf{t}_{\text{target}} = \begin{cases} \text{clip}(\mathbf{o} + \eta,\, 0,\, 1), & \eta \sim \mathcal{N}(\mathbf{0},\, 0.05^2\mathbf{I}),\ \text{w.p. } 0.7, \\ \mathbf{t}_{\text{target}} \sim \mathcal{U}[0,1]^3, & \text{w.p. } 0.3. \end{cases} \tag{3.23}$$

The noise on the in-distribution branch ensures the target almost never equals the achieved outcome, closing the invertible shortcut, while the uniform branch broadens coverage to targets the labelled site does not itself realise, so the model learns to respond to targets across the whole range. At inference the user supplies the target directly.

3.3.9 Training Objective and Protocol

The decoder is trained by teacher forcing — fed the true previous zone at each step — to minimise a masked, per-room cross-entropy. Two refinements shape the loss: padded positions are masked out, and because the contrastive arrangement types are unevenly represented (the single *real* arrangement is far rarer than the five *random* ones), each example is weighted by the inverse square root of its arrangement-type frequency, normalised to unit mean. With w_b the bucket weight and $m_{b,t}$ the room mask,

$$\mathcal{L}_{\text{zone}} = \frac{\sum_b \sum_t w_b\, m_{b,t} \big[-\log P(z^\star_{b,t} \mid z^\star_{b,<t}, \mathcal{G}_b, \mathbf{c}_b) \big]}{\sum_b \sum_t w_b\, m_{b,t}}, \qquad w_b \propto \frac{1}{\sqrt{\text{freq}(b)}}. \tag{3.24}$$

Training runs for 100 epochs with the Adam optimiser (initial learning rate 5×10^{-4}, halved every 30 epochs), a batch size of 128, and gradient-norm clipping at 5.0; plans are capped at 20 rooms. The train, validation, and test splits are formed by *plan identity*: all arrangements and all D_4 transforms of a plan are confined to one split, so evaluation is always on unseen plans. The model has on the order of 0.8 million trainable parameters.

3.3.10 Learned Embedding Structure

Before the trained encoder is used for inference, the learned room embeddings are inspected to verify that the representation is suitable for the conditioned decoder. The graph-attention encoder described in Section 3.3.5 and Equation (3.17) maps each room to a 128-dimensional embedding. To examine this embedding space, Figure 3.12 projects the embeddings into three dimensions using two complementary visualizations.

The first visualization uses t-SNE and colours the projected points by room type. This provides a qualitative check of whether rooms with similar functional roles are placed near one another in the learned representation. The second visualization focuses on living-room embeddings only, holding room type fixed, and colours the same projected points by achieved AQI and QVCI values. This checks whether variation in environmental performance is also reflected within the embedding space.

These visualizations are used as diagnostic checks rather than as quantitative evaluation metrics. Their purpose is to confirm that the encoder representation contains both room-identity information and performance-related variation before it is passed to the autoregressive decoder. This supports the methodological assumption that the decoder can use the encoded room features, together with the conditioning targets, to generate spatial-zone assignments.

3.3.11 Inference

At inference the encoder is run once and the decoder unrolls in free-running mode: at each step it samples a zone from the predicted distribution (Equation (3.20)) and feeds that sample back as the "previous zone" for the next step, while the budget context updates its rooms-remaining and zone-usage entries. Sampling uses a temperature that trades determinism against diversity; several rollouts per plan produce a set of distinct candidate assignments, all conditioned on the same user-supplied target, which are passed to the connectivity and schematic stages.

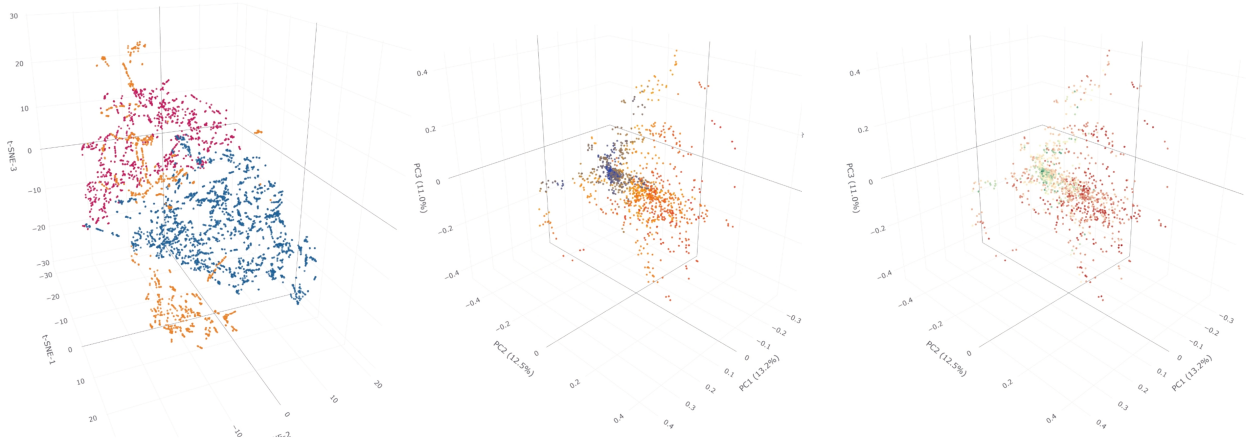


Figure 3.12: Learned room embeddings from the Stage-2 graph-attention encoder, projected to three dimensions. Left: t-SNE projection of all room embeddings coloured by room type; living rooms (pink), bedrooms (blue), and kitchens (orange) form distinct clusters. Middle and right: PCA projection of the living-room embeddings only, recoloured by achieved AQI (middle) and QVCI (right); the smooth colour gradient shows that, with type held fixed, embeddings are ordered by environmental performance. Axis labels on the PCA panels report the variance explained by each component.

3.4 Stage 3 — Connectivity Prediction

The third stage predicts which rooms are connected, producing the room-adjacency graph used to position rooms relative to one another and to the building entrance.

3.4.1 Dataset Preparation for Connectivity Prediction

Training targets are the undirected room-adjacency matrices of the plans, with a single *outside* node added to every graph to represent the entrance. Each graph is stored as its strict lower-triangular adjacency together with the ordered list of node room types; padding extends every graph to the maximum node count observed in the corpus so that batches are rectangular.

3.4.2 Graph Representation and Node Ordering

The nodes are placed in a fixed canonical order — habitable and circulation types first in a preferred order, with the outside node always last — so the autoregressive decomposition below is well defined and consistent with the training graphs. Because the adjacency is symmetric, only the strict lower triangle is independent and needs to be predicted.

3.4.3 Edge-Sequence Formulation

Graph generation is reduced to a sequence of binary decisions, one per possible edge, in lower-triangular order. For n ordered nodes the edge positions are

$$\mathcal{E}_{\text{order}} = [(i, j) : i = 1, \dots, n-1;\ j = 0, \dots, i-1], \qquad |\mathcal{E}_{\text{order}}| = \frac{n(n-1)}{2}, \tag{3.25}$$

and at each position the model emits one bit $b_{ij} \in \{0, 1\}$. This turns graph generation into a sequence-labelling problem of length $n(n-1)/2$.

3.4.4 Autoregressive Edge Model

Each edge decision reads the types of the two endpoint rooms and the previous decision. Room types map to learned embeddings of dimension 128 and the previous edge bit to an embedding of dimension 16; their concatenation ($2 \times 128 + 16 = 272$) is the per-step input. A single-layer gated recurrent unit (Cho et al. 2014) of hidden width 512 carries the graph state, and a linear head produces the edge logit:

$$\mathbf{r}_{ij} = [\, \mathbf{e}_{t_i} \,\|\, \mathbf{e}_{t_j} \,\|\, \mathbf{e}_{b_{\text{prev}}} \,], \tag{3.26}$$

$$\mathbf{q}_{ij} = \text{GRU}(\mathbf{r}_{ij},\, \mathbf{q}_{\text{prev}}), \qquad p_{ij} = \sigma(\mathbf{w}^\top \mathbf{q}_{ij} + b), \tag{3.27}$$

with σ the logistic function. The graph distribution factorises autoregressively in the spirit of You et al. (2018b),

$$P(\mathbf{A}) = \prod_{0 \le j < i < n} P(b_{ij} \mid b_{<(i,j)},\, \mathbf{t}). \tag{3.28}$$

The recurrent state summarises the whole partial graph, so each new edge is informed by every edge already laid down — reproducing patterns such as a kitchen and living room sharing an opening, or a bathroom connecting to a single circulation node. The model has no separate encoder: the room-type embeddings are a lookup, and the recurrent state alone carries the graph context, so a single GRU layer suffices. Figure 3.13 shows the edge model, and Figure 3.14 its gated recurrent cell.

3.4.5 Training Objective

The model is trained by teacher forcing with the true previous edge bit. The loss is a masked binary cross-entropy over the real edge positions (positions beyond $n(n-1)/2$ are padding and masked),

$$\mathcal{L}_{\text{conn}} = -\,\frac{\sum_t m_t [b_t \log p_t + (1 - b_t) \log(1 - p_t)]}{\sum_t m_t}, \tag{3.29}$$

Stage 3 — Connectivity Model (Autoregressive Edge GRU)

Per-step computation for one edge (i, j): 272 → GRU(512) → logit → σ, decoded in lower-triangular order.

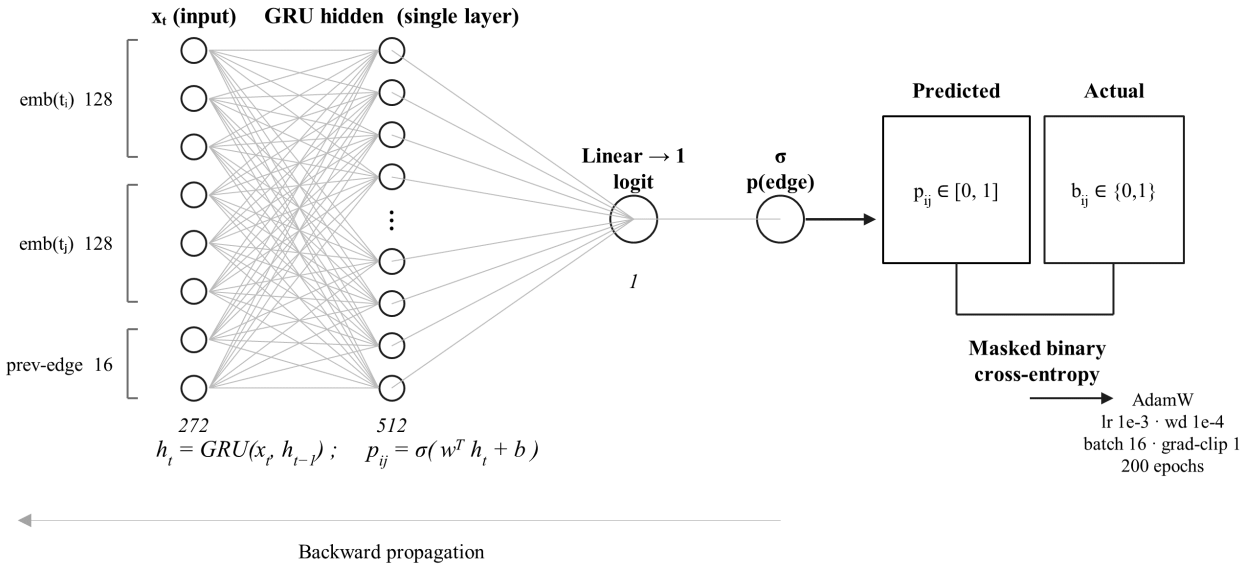


Figure 3.13: The connectivity model as a layered network. The room-type embeddings of an edge's two endpoints and the previous edge bit feed a single-layer GRU, whose state is read out by a linear-plus-sigmoid head to an edge probability; edges are decoded in lower-triangular order.

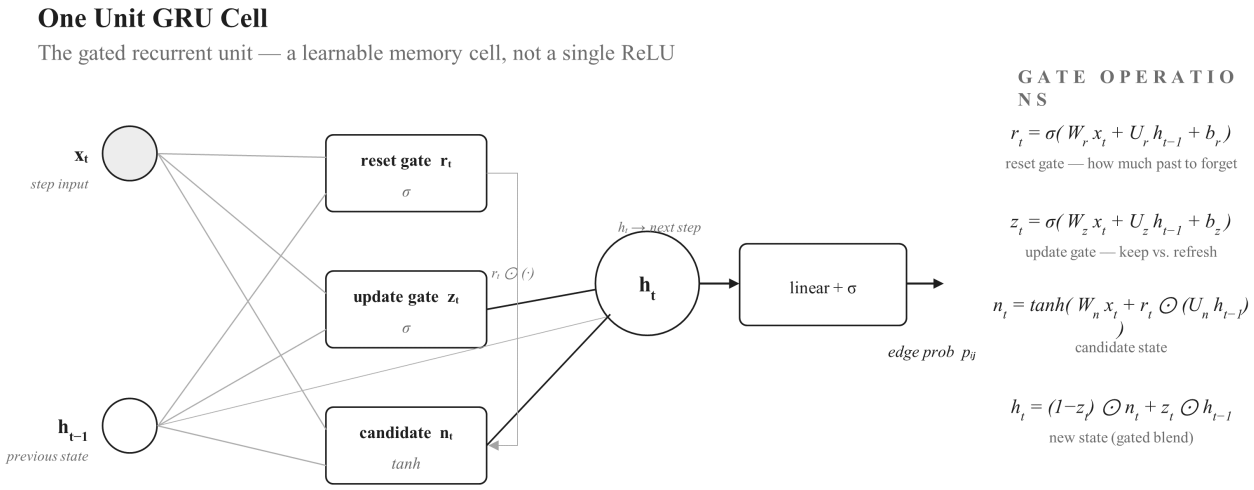


Figure 3.14: Anatomy of the gated recurrent unit used in the connectivity model — the same gated cell as in the zone decoder, here a single layer of width 512 — with a linear-plus-sigmoid head producing each edge probability.

with m_t the position mask. Training runs for 200 epochs with the AdamW optimiser (learning rate 10^{-3}, weight decay 10^{-4}), a batch size of 16, and gradient-norm clipping at 1.0.

3.4.6 Inference and Graph Assembly

At inference the edges are decoded in the same order, but each bit is realised by *Bernoulli sampling* of its probability, $b_{ij} \sim \text{Bernoulli}(p_{ij})$, rather than thresholding. Sampling is preferred because a hard threshold discards borderline edges that can leave a primary room isolated, whereas sampling admits them at their learned frequency and reproduces the training connectivity statistics; a fixed seed keeps the output reproducible. The graph is then laid over the schematic: primary rooms are pinned at the centroids of their Stage-2 zone cells, and the remaining rooms and the outside node are relaxed by a spring layout against those anchors, so connected rooms settle adjacent.

3.5 Stage 4 — Schematic Synthesis and Pareto Selection

The final stage assembles the predictions into concrete geometry and selects the best layouts. Each room's predicted area (Stage 1), aspect ratio, and WWR fix the dimensions of a rectangle; its predicted zone (Stage 2) anchors the rectangle in the 3×3 grid; and the predicted connectivity (Stage 3) constrains relative placement so connected rooms are adjacent, yielding a schematic rectangular layout.

Because one set of inputs can be realised as many layouts, and because QVCI and AQI are in tension (Section 3.3.2.3), the stage produces a population of candidates and evaluates each one's $(\text{QVCI}, \text{AQI})$ pair. A layout ℓ Pareto-dominates ℓ' when it is at least as good on both objectives and strictly better on one,

$$\ell \succ \ell' \iff \text{QVCI}(\ell) \geq \text{QVCI}(\ell') \ \wedge \ \text{AQI}(\ell) \geq \text{AQI}(\ell') \ \wedge \ \ell \neq \ell' \text{ strictly on one,} \tag{3.30}$$

and the Pareto front is the set of non-dominated layouts,

$$\mathcal{F} = \{\, \ell : \nexists\, \ell' \text{ with } \ell' \succ \ell \,\}. \tag{3.31}$$

It is this front, not any single layout, that is presented to the designer: each point cannot be improved on one metric without sacrificing the other.

3.5.1 Force-Directed Schematic Assembly

The zone grid determines the zone assigned to each room, but it does not define the room's exact rectangular shape or its precise contact with neighbouring rooms. Stage 4 resolves this by using a short force-directed assembly process. Importantly, Stage 4 does not modify the zone assignments produced in Stage 3.3. These assignments are treated as fixed inputs to the schematic assembly process. Stage 4 only resolves the rectangular dimensions of each room and adjusts local room-to-room contact within the predicted zone arrangement. In this way, the process produces a compact, non-overlapping schematic layout while preserving all zone assignments from the preceding stage.

Each room is first represented as a rectangle. Its shape is determined from the predicted room area A (Stage 3.2) and a room-type aspect ratio a, clamped to the range $[0.3, 3.0]$, with height and width defined as

$$h = \sqrt{A/a}, \qquad w = a\,h.$$

The centre of each rectangle is initialized from the room's position in the Stage 3.4 connectivity graph and then mapped into the drawing boundary. This initialization fixes the relative zone arrangement, while the subsequent relaxation only adjusts local position and contact.

The relaxation process combines repulsive and attractive forces. First, overlapping rectangles are separated along the direction of least penetration. For two rectangles whose centres differ by $(\Delta x, \Delta y)$, the horizontal and vertical overlaps are

$$o_x = \frac{w_a + w_b}{2} - |\Delta x|, \qquad o_y = \frac{h_a + h_b}{2} - |\Delta y|.$$

If both overlaps are positive, the smaller overlap is removed by moving the two rectangles apart. After each move, the rectangles are clamped inside the padded drawing boundary.

Attractive forces are then applied to keep the layout compact and to bring connected rooms into contact. A weak centroid force moves each rectangle slightly toward the centre of the group. In addition, each adjacency predicted in Stage 3.4 is treated as a spring. For a connected pair of rooms, the target centre-to-centre distance at which the rectangles just touch is

$$d^\star = |\hat{u}_x|\,\frac{w_a + w_b}{2} + |\hat{u}_y|\,\frac{h_a + h_b}{2}, \tag{3.32}$$

where $(\hat{u}_x, \hat{u}_y)$ is the unit direction between the two room centres. If the actual distance d is greater than $d^\star$, the rooms are pulled together with stiffness $k = 0.3$. The spring is inactive once the connected rooms

are in contact, so adjacency is encouraged without forcing overlaps.

The assembly process repeats the centroid force, adjacency springs, and overlap-resolution steps for thirty cycles. Finally, the packed room cluster is translated as a rigid group so that one edge aligns with the requested facade direction. The result is a schematic rectangular layout in which rooms are sized by predicted area, placed according to predicted zones, and arranged so that predicted adjacencies are represented by physical contact.

3.6 Implementation and Integrated Tool

The four stages are implemented through web-based tools and a desktop application that guide the designer from program inputs through area prediction, zone assignment, and connectivity prediction to final schematic layouts and Pareto-front selection. The trained models run behind a consistent interface, allowing conditioning targets to be adjusted interactively and their consequences to be inspected stage by stage. Together, these tools realise the performance-as-condition workflow that motivates the thesis.

3.7 Experimental Setup and Evaluation

Training regime. The models were trained on a single consumer GPU (an RTX 3060 laptop-class device). The zone model converges over roughly one hundred epochs using the plan-level splits of Section 3.3.9; the area and connectivity models train comparably quickly.

Predictive accuracy. The primary measure for the zone model is teacher-forced classification accuracy on held-out test plans, reported overall and by arrangement type and room type, with a 9×9 zone confusion matrix. The connectivity model is evaluated by its masked edge cross-entropy on held-out graphs.

Conditioning effectiveness. Each conditioning axis is swept from its low to its high setting while the others are held at midpoint, and the movement in the achieved metric is measured as a directional gap,

$$\Delta_{\mathrm{QVCI}} = \overline{\mathrm{QVCI}}\big|_{0.9} - \overline{\mathrm{QVCI}}\big|_{0.1}, \tag{3.33}$$

$$\Delta_{\mathrm{AQI}} = \overline{\mathrm{AQI}}\big|_{0.9} - \overline{\mathrm{AQI}}\big|_{0.1}, \tag{3.34}$$

$$\Delta_{\mathrm{spread}} = \overline{\mathrm{Spread}}\big|_{0.9} - \overline{\mathrm{Spread}}\big|_{0.1}, \tag{3.35}$$

all expected to be substantially positive. Monotonicity is confirmed by checking that achieved values rise as a target is swept and that a balanced target lands between the extremes, visualised as a target–outcome

surface.

Representation analysis. Finally, the learned room embeddings (zone model) and graph embeddings (connectivity model) are projected to low dimension to inspect the structure the encoders have learned — clustering by room type, zone, performance outcome, and node count.

3.7.1 Evaluation Setup

To evaluate the trained model, a single representative plan configuration is held fixed while the two performance targets are varied. Holding every other input constant isolates the model's response to the conditioning signal, so that any change in the generated layouts can be attributed to the requested targets rather than to differences in the plan or its site. The fixed configuration is summarised in Table 3.2.

The plan comprises ten rooms on a 1500×2000 px boundary (3,000,000 px^2), with the room counts, per-type aspect ratios, and per-type window-to-wall ratios listed in Table 3.2. The site context is asymmetric: the north faces vegetation and the west faces open sky, both favourable for view, while the south and east face no view and carry the highest outdoor sound pressure levels (82 and 85 dB). This asymmetry is deliberate, as it gives the model a non-trivial trade-off to resolve when the view and acoustic targets pull in different directions.

Against this fixed backdrop, only the view and acoustic targets are varied. The target Quality View Coverage Index and target Acoustic Quality Index are each swept across their range, while the spread target is held constant. Each target combination is supplied to the generator, and the resulting layouts are evaluated as described in Section 3.2. Sweeping the two targets over a grid in this way traces the model's response surface and exposes the view–acoustic trade-off directly.

Table 3.2: Fixed plan configuration used for evaluation. Only the view and acoustic targets are varied; all values below are held constant.

Room counts (10 rooms total)			
Living	1	Bedroom	3
Kitchen	1	Bathroom	2
Balcony	2	Storage	1
Plan geometry			
Width (E–W)	1500 px	Height (N–S)	2000 px
Total area	3,000,000 px^2		
Site conditions (view / outdoor SPL per direction)			
North	Vegetation, 62 dB	South	No view, 82 dB
East	No view, 85 dB	West	Sky, 55 dB
Aspect ratio (per room type)			
Living	1.10	Bedroom	1.00
Kitchen	1.40	Bathroom	0.90
Balcony	2.50	Storage	1.40
Window-to-wall ratio (per room type)			
Living	0.15	Bedroom	0.15
Kitchen	0.20	Bathroom	0.05
Balcony	0.60	Storage	0.03
Targets			
QVCI	*varied*	AQI	*varied*
Spread	1.00 (fixed)		

Chapter 4

Results and Discussion

4.1 Overview

This chapter evaluates the trained pipeline across accuracy, controllability, connectivity, layout variation, and trade-off selection. Section 4.2 reports the prediction accuracy of the spatial zoning model. Section 4.3 evaluates target controllability by testing whether the model responds systematically to changes in performance targets, using the directional-gap measures in Equations (3.33)–(3.35) and the slider-responsiveness analysis. Section 4.4 examines the connectivity stage, which translates zoned room assignments into a room graph. Section 4.5 presents generated schematic layouts as the performance targets are swept. Section 4.6 then shows how the Pareto front in Equation (3.31) supports the selection of design trade-offs. Finally, Section 4.7 brings these findings together and identifies the limitations of the pipeline.

4.2 Predictive Performance of the Zone Model

The performance-conditioned spatial zoning model achieved approximately 86% accuracy in predicting room-zone assignments, with a masked cross-entropy loss of 0.20 (Equation (3.24)). Since the goal of the model is not to reproduce a single reference arrangement, but to generate layouts that satisfy requested performance targets, target prediction accuracy is also reported. This measure evaluates whether the achieved performance falls within the correct target band. The model reached approximately 94% target prediction accuracy, with a root-mean-square error of 0.06 between requested and achieved values. This difference between assignment accuracy and target accuracy is expected because multiple room-zone configurations can produce similar plan-level QVCI, AQI, and spread values. Therefore, the model may differ from a reference zone assignment while still achieving the intended performance targets. Table 4.1 summarizes these results.

Table 4.1: Predictive performance of the zone model on the held-out test set.

Measure	**Value**
Zone classification accuracy	$\approx 86\%$
Target prediction accuracy	$\approx 94\%$
Cross-entropy loss	0.20
Root-mean-square error (target vs. achieved)	0.06

4.3 Conditioning Effectiveness

Accuracy alone does not demonstrate that the model is controllable. To evaluate controllability, each conditioning target was varied from its low setting to its high setting while the other two targets were held at their midpoint. The resulting change in the achieved performance metric was measured using the directional-gap measures defined in Section 3.7. All three gaps are positive and exceed the working threshold of 0.20:

$$\Delta_{\text{QVCI}} = +0.71, \qquad \Delta_{\text{AQI}} = +0.54, \qquad \Delta_{\text{spread}} = +0.66. \tag{4.1}$$

Slider responsiveness provides a second measure of controllability. It measures the fraction of slider changes that move the achieved metric in the intended direction. The responsiveness values are approximately 82% for Quality View, 78% for spread, and 67% for acoustic performance. These results show the same overall pattern as the directional gaps: the Quality View target is the easiest to control, the spread target is intermediate, and the acoustic target is more variable (Table 4.2).

The acoustic result confirms that the model can also respond to acoustic-performance targets. Although the acoustic axis is less monotonic than the Quality View axis, a directional gap of +0.54 and a responsiveness value of 67% indicate a clear and meaningful shift in achieved AQI. This suggests that the model captures the influence of acoustic conditioning while reflecting the greater complexity of acoustic interactions in spatial layouts. The target sweep in Section 4.5 further demonstrates that achieved AQI can be steered across most of its range as the acoustic target is varied.

Table 4.2: Conditioning effectiveness by axis. Gaps are achieved-metric movement between the low (0.1) and high (0.9) target settings; responsiveness is the approximate fraction of slider movements producing the intended direction of change.

Conditioning axis	**Directional gap**	**Slider responsiveness**
Quality view (QVCI)	+0.71	$\approx 82\%$
Acoustic (AQI)	+0.54	$\approx 67\%$
Spread	+0.66	$\approx 78\%$

The more variable response of the acoustic axis is not unexpected; rather, it reflects the coupled nature of Quality View and acoustic performance discussed in Section 3.3.2.3. The exposed perimeter zones that

support higher Quality View scores can also admit more outdoor noise (Equation (3.11)). Therefore, when high Quality View and high acoustic-performance targets are requested together, the model must negotiate a genuine spatial trade-off. The acoustic response remains directionally effective, but its smoother control is limited by this underlying competition between visual access and noise exposure.

4.4 Connectivity Generation

Once a plan has been zoned, the Stage-3 connectivity model predicts which rooms are joined (Section 3.4.4). Figure 4.1 shows the generated room graphs for a batch of ten-room plans. The graphs are consistent and architecturally plausible: each plan is fully connected by eleven edges — a spanning structure with a small number of additional loops — and the kitchen and living spaces consistently emerge as high-degree hubs while bathrooms and balconies attach as leaves. This is the topology one expects of a residential unit, where circulation concentrates on the social rooms and service rooms sit at the periphery, and it confirms that the connectivity stage produces usable adjacency for the schematic synthesis that follows.

4.5 Generated Design Variations

To illustrate the generative behaviour, a representative ten-room plan (3 000 000 area units(px), spread target 1.0) was passed through the full pipeline and its performance targets were swept. Each run returns twelve candidate layouts, ranked by how closely they match the requested targets; the contact sheets below show the top nine of the twelve as complete schematics — rooms placed in their predicted zones, sized by their predicted areas, and annotated with the achieved plan metrics QVCI (Q) and AQI (A).

4.5.1 Baseline targets

Figure 4.2 shows the candidates at the neutral setting, with target QVCI = 0.50 and AQI = 0.50. Table 4.3 lists their achieved metrics. Two observations stand out. First, the population spans a wide performance range, with achieved QVCI from 0.40 to 0.76 and AQI from 0.23 to 0.51. This indicates that, even at a single target setting, the pipeline returns varied layout alternatives rather than near-duplicates. Second, at this setting, QVCI and AQI are positively correlated across the candidates. The highest-view candidate (#01, QVCI = 0.755) also achieves the highest AQI (0.507), while the lowest-view candidates also have lower AQI values. Given the input site conditions for this generated case, the side with better acoustic exposure also corresponds to the side with stronger view access. Therefore, at the neutral target setting, QVCI and AQI are expected to align rather than compete. The observed positive relationship between view and acoustic

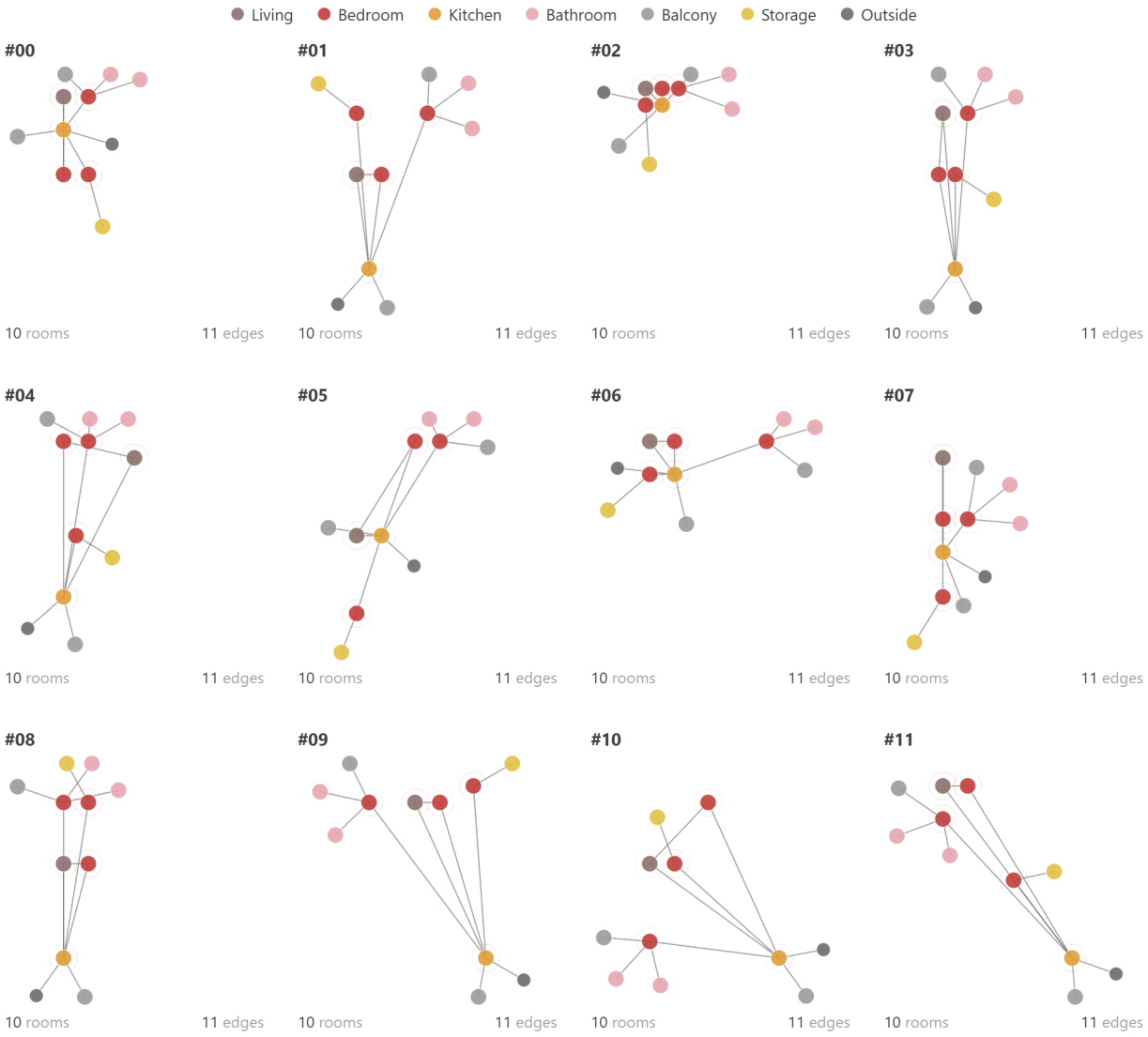


Figure 4.1: Generated room-connectivity graphs for a batch of ten-room plans produced by the Stage-3 connectivity model. Node colour encodes room type (legend at top); each plan has ten rooms joined by eleven edges. Kitchen and living spaces appear as high-degree hubs, with bathrooms and balconies as leaves.

performance is thus consistent with the generation conditions supplied to the model.

Table 4.3: Achieved metrics for the nine displayed candidates of Figure 4.2 (target QVCI = 0.50, AQI = 0.50), ordered by achieved QVCI.

Candidate	Achieved QVCI	Achieved AQI
#01	0.755	0.507
#02	0.614	0.378
#03	0.614	0.407
#04	0.614	0.407
#05	0.473	0.307
#06	0.473	0.307
#09	0.434	0.252
#08	0.403	0.254
#07	0.403	0.225

4.5.2 Raising the acoustic target

Holding the view target at 0.50 and raising the acoustic target to 0.70 produces the candidates shown in Figure 4.3. The achieved AQI of the leading candidates increases substantially: candidate #01 reaches AQI 0.685 and candidate #02 reaches AQI 0.665, compared with a best value of 0.507 at the neutral target. At the same time, a high-view option remains available, with candidate #04 achieving QVCI 0.760 and AQI 0.531.

These results show that the acoustic target is not fixed. When a higher acoustic target is requested, the pipeline produces layouts with higher achieved AQI. The adjustment involves a moderate trade-off with view performance rather than a strict performance ceiling. The full target sweep in the next section quantifies this behaviour across the complete target range, while the remaining contact sheets for the intermediate and extreme targets are provided in Appendix D.

4.6 The Pareto Front and Design Selection

The generated population provides a range of possible design alternatives, but the strongest compromises can be identified through Pareto selection. To do this, the full population was plotted in the QVCI–AQI plane and evaluated using the dominance relation defined in Equation (3.30). This process identifies the non-dominated front: the set of layouts that offer the most efficient balance between the two performance metrics.

Figure 4.4 shows this comparison for two target settings. At the neutral setting, where both targets are 0.50, the efficient front extends diagonally upward, reaching approximately QVCI 0.86 and AQI 0.60.

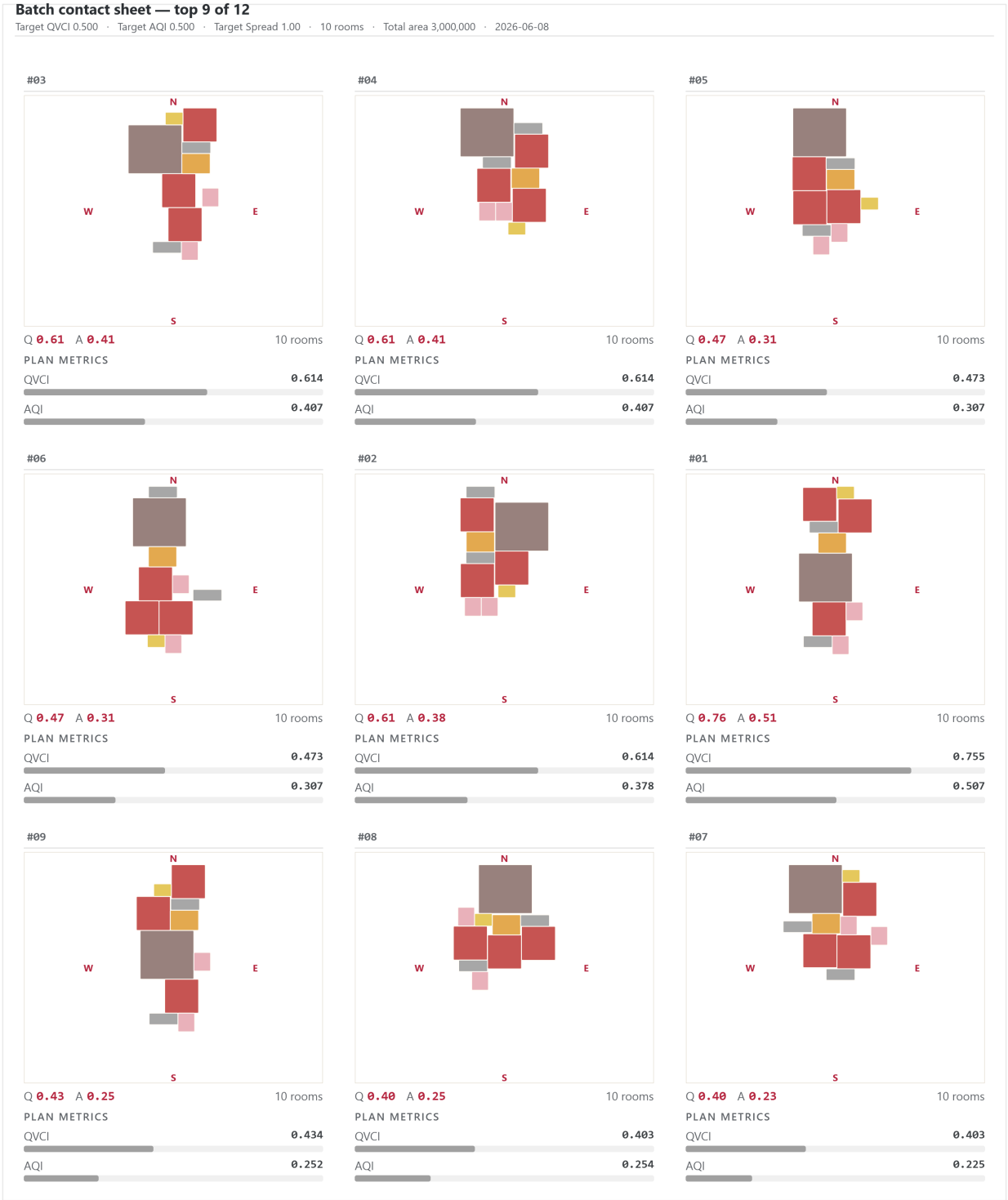


Figure 4.2: Generated schematic layouts for the representative ten-room plan at target QVCI = 0.50, AQI = 0.50 (spread = 1.0). The top nine of twelve candidates are shown; each panel reports its achieved QVCI (Q) and AQI (A) and the corresponding metric bars.

Figure 4.3: Generated schematic layouts for the same plan with the acoustic target raised to AQI = 0.70 (view target held at QVCI = 0.50). The achieved AQI of the leading candidates rises to as high as 0.685, while a high-view candidate (QVCI 0.760) is still produced.

When both targets are raised to 1.00, the generated population shifts toward the high-AQI range, approximately 0.80–1.00, and the Pareto front is selected from this higher-performing region. The gray points represent additional generated candidates, while the highlighted front identifies the alternatives that provide the strongest performance trade-offs.

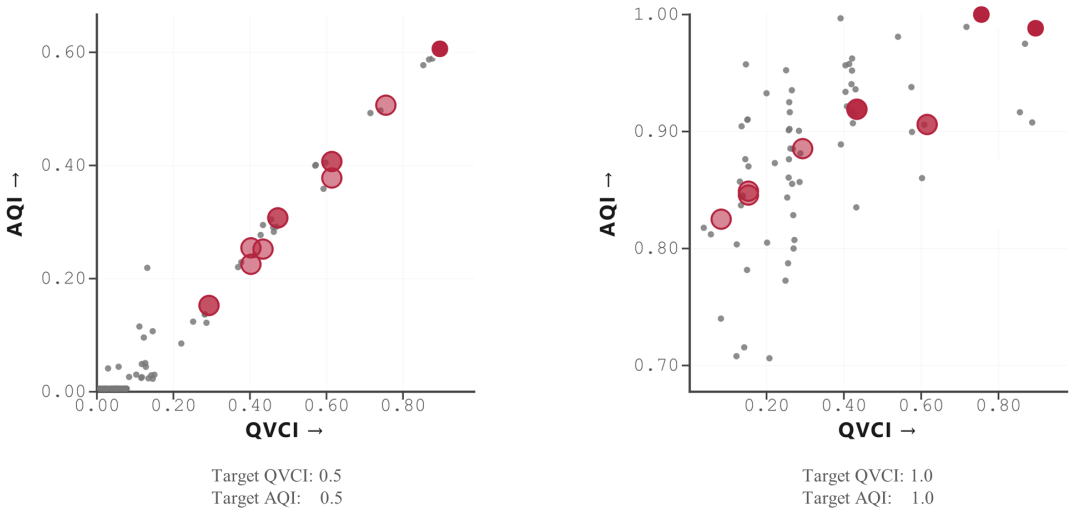


Figure 4.4: Generated populations in the QVCI–AQI plane for two target settings: both metrics at 0.5 (left) and both at 1.0 (right). The highlighted points are the non-dominated (Pareto-efficient) layouts; the small grey points are dominated.

The systematic sweep in Figure 4.5 shows how the two conditioning axes affect the generated population and how they interact. In the left column, the view target is held at 0.50 while the acoustic target is increased from 0.60 to 1.00. As the acoustic target increases, the efficient front moves upward, with the achieved AQI of the leading layouts rising from approximately 0.40 to nearly 1.00, while QVCI remains within a moderate range. In the right column, the acoustic target is held at 0.50 while the view target is increased from 0.60 to 1.00. In this case, the efficient front moves to the right, with achieved QVCI approaching 1.00, while achieved AQI decreases toward approximately 0.20–0.40. These results show that both performance axes are controllable across most of their ranges.

The sweep also shows that the interaction between QVCI and AQI is asymmetric. Increasing the view target to its highest setting reduces acoustic performance more strongly than increasing the acoustic target reduces view performance. This pattern is consistent with the spatial logic of the metrics: high Quality View is mainly achieved in exposed perimeter zones, while acoustic quality can also be improved by assigning rooms to more sheltered interior zones.

This result demonstrates the value of treating performance as a condition of generation. Rather than producing a single layout and evaluating it afterward, the pipeline generates a population of layouts already organized around the QVCI–AQI trade-off. The Pareto front then identifies the most efficient alternatives

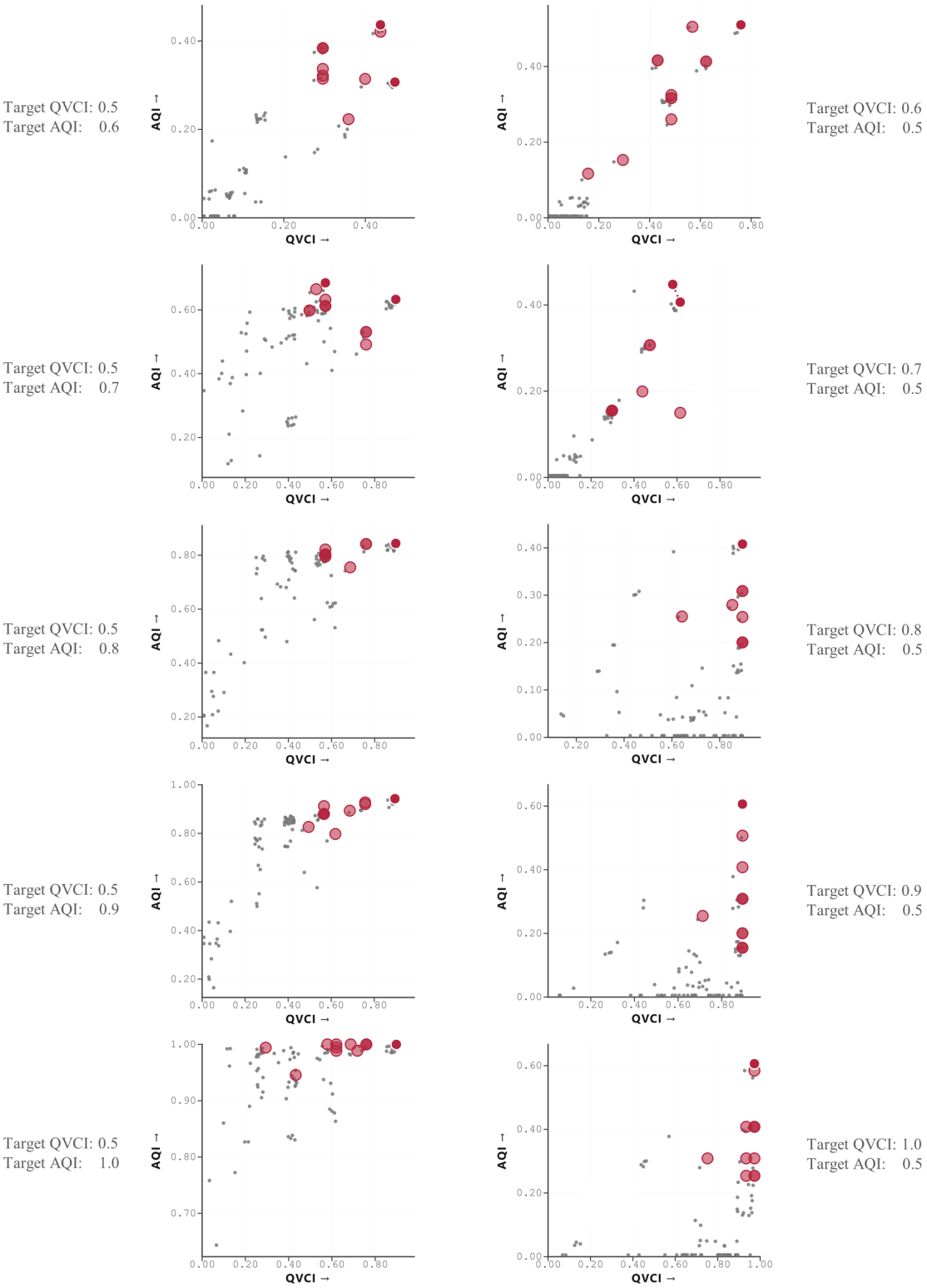


Figure 4.5: Systematic target sweep in the QVCI–AQI plane. Left column: view target fixed at QVCI = 0.5, acoustic target increased from AQI = 0.6 to 1.0 (top to bottom). Right column: acoustic target fixed at AQI = 0.5, view target increased from QVCI = 0.6 to 1.0. Highlighted points are the Pareto-efficient layouts at each setting.

within that population. These alternatives do not reduce the design decision to a single numerical optimum; instead, they present a smaller set of strong compromise layouts, allowing the designer to consider spatial qualities that are not fully captured by the metrics.

4.7 Discussion

Taken together, the results support the central claim of this thesis. The spatial zoning model performs well in terms of prediction accuracy, achieving approximately 86% zone-assignment accuracy with a cross-entropy loss of 0.20. More importantly, the model demonstrates target controllability. Sweeping each target produces a large and correctly directed change in the corresponding achieved metric, with $\Delta_{\mathrm{QVCI}} = +0.71$, $\Delta_{\mathrm{AQI}} = +0.54$, and $\Delta_{\mathrm{spread}} = +0.66$. The high target-prediction accuracy of approximately 94%, with an RMSE of 0.06, further indicates that requested performance values are reliably achieved. Figure 4.5 confirms this visually for both environmental axes: achieved AQI can be increased from approximately 0.40 to nearly 1.00, and achieved QVCI can also approach 1.00 as the corresponding target is raised. These results suggest that the conditioning strategy described in Section 3.3.8 is effective, since the model responds to target values across a broad range rather than simply reproducing memorized arrangements.

The results also identify several limitations. The acoustic axis is controllable, but it is less smooth than the view axis, with 67% slider responsiveness compared with 82% for Quality View. The QVCI–AQI trade-off is also asymmetric: increasing the view metric to its highest range reduces acoustic quality more strongly than the reverse case. This reflects the spatial relationship discussed in Section 3.3.2.3, where high Quality View is associated with exposed perimeter zones that can also admit higher outdoor noise (Equation (3.11)). A second limitation concerns generality. The layout variations examined here are based on a single representative ten-room plan, and the relationship between QVCI and AQI depends on site conditions. In some cases the two metrics may align, as shown at the neutral target setting in Table 4.3; in other cases they may compete, as shown in the high-target sweep. Finally, the reported accuracy and conditioning values are point estimates from the trained model evaluation. A broader validation study could report confidence intervals across multiple plans, target settings, and random seeds.

These limitations do not weaken the main finding. Across a physically coupled pair of environmental objectives, the pipeline generates diverse and plausible layouts whose measured performance responds to user-defined targets. The Pareto front then identifies the strongest trade-off alternatives within the generated population. This demonstrates the core capability proposed by the framework: treating environmental performance not as a post-generation check, but as a condition that guides layout generation.

Chapter 5

Conclusion and Future Work

5.1 Summary of the Research

This thesis addresses a persistent gap in computational layout generation. In most existing workflows, environmental performance is evaluated after a layout has already been produced. As a result, achieving a performance goal often requires repeated cycles of simulation and revision. In addition, many learning-based floor-plan generation methods represent plans as images, which can be computationally demanding and does not explicitly encode environmental performance. In response, this thesis proposes a scalable, prediction-based framework that treats quantified environmental performance as a condition for generation, rather than as a property measured afterward.

The framework decomposes layout generation into four sequential prediction stages: area distribution, zone assignment, connectivity prediction, and schematic synthesis with Pareto selection (Chapter 3). Two environmental performance indicators guide the pipeline: the Quality View Coverage Index (QVCI) and the Acoustic Quality Index (AQI). These indicators were selected because both are affected by the window-to-wall ratio and by the position of rooms within the site. They also represent a meaningful trade-off, since the exposed zones that support stronger Quality View can also admit higher levels of outdoor noise. A spatial-spread target is included as an additional conditioning axis to guide the distribution of rooms across the plan.

The main learning component is a performance-conditioned spatial zoning model. It uses a graph attention encoder to produce site-aware room embeddings and an autoregressive decoder to assign each room to a spatial zone while reading the performance targets during generation. The model is trained on a contrastive corpus that pairs each plan with multiple arrangements of different performance levels. This allows

the model to learn how spatial assignments affect performance, rather than simply reproducing one fixed reference layout.

The results in Chapter 4 show that the trained pipeline is both accurate and controllable. The spatial zoning model achieves approximately 86% zone-assignment accuracy, while the target-sweep results show strong controllability across the conditioning axes. Sweeping each target produces a large and correctly directed change in the achieved metric, with $\Delta_{\mathrm{QVCI}} = +0.71$, $\Delta_{\mathrm{AQI}} = +0.54$, and $\Delta_{\mathrm{spread}} = +0.66$. The model also achieves approximately 94% target-prediction accuracy, with a root-mean-square error of 0.06 between requested and achieved values. Finally, Pareto selection organizes the generated population into a smaller set of strong trade-off alternatives. Together, these results demonstrate the central contribution of the thesis: environmental performance can be used as a condition that guides layout generation, rather than only as a post-generation evaluation step.

5.2 Revisiting the Research Questions

The study was guided by the two research questions posed in Section 1.6.

Research Question 1. *How can architectural layouts be generated using a prediction-based machine learning framework that is conditioned on environmental performance metrics during—rather than after—the generation process?* The framework answers this by carrying the performance targets inside the generative loop. The zone decoder receives a budget context at every step that includes the target QVCI, AQI, and spread, so each room is placed with the goal in view rather than having its performance checked once the plan is complete. A 70/30 target-sampling rule during training keeps this conditioning genuine instead of memorised, by ensuring the requested target rarely equals the achieved outcome of the labelled arrangement. The evidence that this works is direct: the strongly positive directional gaps of Equation (4.1) and the high target accuracy show that the generated layouts move on command toward the requested performance, including values a given plan does not itself realise.

Research Question 2. *How can layouts be generated from structured geometric data rather than image-based representations, so that the generation process remains computationally lightweight and explicitly aware of environmental performance?* The framework represents a plan as a graph of rooms, each carrying a compact feature vector of geometry, view, acoustic, and derived performance quantities, together with plan-level zone-weight vectors — not as pixels. This keeps the model small (on the order of 0.8 million parameters, trainable on a single consumer laptop GPU) and makes performance an explicit part of the representation

rather than something inferred from an image: the conditioning metrics are computed from the structured geometry itself, and the same structured features feed both generation and evaluation. The framework therefore generates layouts that are at once lightweight to produce and performance-aware by construction.

5.3 Key Contributions

Against the contributions stated in Section 1.7, the work delivers the following.

- **A prediction-based layout generation framework** — realised as the four-stage, individually conditionable pipeline of Chapter 3, which inverts the conventional generate-then-simulate order.
- **A method for integrating environmental performance metrics into generation** — the QVCI and AQI metrics serve as conditioning targets, and the contrastive training corpus, together with the leakage-avoiding target-sampling rule, is the means by which the model learns to honour them and to navigate their trade-off.
- **A demonstrative residential test case** — the experiments of Chapter 4, in which the pipeline generates and ranks candidate layouts for residential apartment units.
- **A scalable, extendable framework** — the model handles plans of varying size and room count in a single pass, and the conditioning channel is extendable: the structural spread axis already shows that a further conditioning dimension can be added without changing the architecture, anticipating additional metrics in future work.
- **A workflow that reduces reliance on repeated post-generation simulation** — because the pipeline produces a spectrum of layouts already positioned along the QVCI–AQI trade-off and surfaces the efficient ones through the Pareto front, the designer is given a set of performing options up front rather than one layout to simulate and revise.
- **An interactive application** — an integrated tool that walks a designer from program inputs through the staged predictions to the final schematics and their Pareto front, offering a useful number of layout options at the start of a project.

Taken together, these shift the emphasis from automatic space allocation toward performance-informed layout generation, strengthening the link between machine learning and early-stage architectural decision-making.

5.4 Limitations

The framework's limitations are of two kinds: those inherent in the scope chosen for this study, and those the results themselves exposed.

The scope limitations follow from the deliberate bounding of the problem (Section 1.5). The framework is demonstrated on two environmental metrics, quality view and acoustic performance, rather than on the full range of environmental factors that shape a layout; daylight, thermal comfort, and ventilation, among others, are not yet modelled. The quality of the generated layouts also depends on the quality, consistency, and diversity of the training data, so limited variation in the source plans can propagate into the outputs. The performance metrics used during generation are study-specific environmental performance indicators developed for machine-learning training and early-stage design exploration; they do not replace detailed environmental simulation or post-design validation.

The results also clarify the expected constraints of the framework, as discussed in Chapter 4. First, the acoustic axis is controllable but less smooth than the view axis, with approximately 67% slider responsiveness compared with 82% for Quality View. This behaviour is consistent with the physical relationship encoded in the framework. High Quality View is typically associated with exposed perimeter zones, while stronger acoustic performance often benefits from more sheltered locations. As a result, when both view and acoustic targets are requested together, the model must negotiate a real spatial trade-off. The positive acoustic directional gap of +0.54 shows that the acoustic target still produces a substantial and meaningful response, even though its range of movement is more constrained than the view axis.

Second, the detailed design variations are based on a single representative ten-room plan. Therefore, the relationship between QVCI and AQI should be understood as site-dependent rather than universal. In some site conditions, the two metrics may align, as observed at the neutral target setting in Table 4.3; in others, they may compete more strongly, as shown in the high-target sweep. This site dependence is expected, because both view access and acoustic exposure are influenced by the orientation and boundary conditions of the site.

Finally, the reported performance metrics are approximate point estimates from the trained model evaluation. A broader validation study could extend this work by reporting confidence intervals across multiple plans, target settings, and random seeds. These points do not weaken the main result; rather, they define the expected scope of the current demonstration and identify clear directions for future validation.

5.5 Future Work

5.5.1 Incorporating Additional Environmental Contexts

The most direct extension is to widen the set of conditioning metrics. Because the budget context is a concatenated vector of targets, adding a new environmental axis — daylight, thermal comfort, or ventilation — requires extending that vector and supplying matching contrastive arrangements, not redesigning the architecture; the spread axis already demonstrates that an additional dimension can be conditioned on cleanly. A related refinement is to improve the acoustic representation itself: a more faithful outdoor-noise and transmission model would sharpen the AQI signal and may partially decouple it from QVCI, easing the trade-off headwind identified in Chapter 4 and giving the acoustic slider more room to move.

5.5.2 Multi-Unit and Mixed-Use Extensions

This study adopted single residential apartment units as its test case. A natural progression is to scale the framework to multiple units sharing a floor plate, with shared circulation and service cores, and ultimately to mixed-use programs that combine residential, commercial, and amenity spaces. These settings introduce inter-unit relationships and shared constraints that the present single-unit formulation does not model, and would test the scalability the framework was designed for at a larger and more heterogeneous scale.

5.5.3 End-to-End Differentiable Pipeline

The four stages are currently trained separately and chained at inference, which keeps each stage simple and inspectable but prevents errors and objectives from being optimised jointly. A promising direction is to make the pipeline end-to-end differentiable, so that gradients can flow from the final, evaluated layout back through connectivity, zoning, and area allocation. This would let the framework optimise the achieved performance of the finished layout directly, rather than optimising each stage against its own intermediate target, and could tighten the agreement between requested and achieved performance still further.

5.6 Closing Remarks

The contribution of this thesis is less a single model than a change of order. By supplying environmental performance as an input target and training generative models to meet it, the framework moves environmental reasoning from the end of the design loop to its beginning, and it does so over a deliberately difficult pair of objectives that pull against one another. The pipeline generates diverse, plausible residential layouts

whose measured performance moves on command and whose best compromises are surfaced automatically through a Pareto front — the capability the framework set out to demonstrate. Quality view and acoustic performance were only a test pair; the more durable result is a structured, lightweight, and extendable way of letting performance lead generation rather than follow it, and a foundation on which further environmental contexts and larger architectural programs can be built.

References

Abouagour, Mohamed and Eleftherios Garyfallidis (2025). "ResPlan: A Large-Scale Vector-Graph Dataset of 17,000 Residential Floor Plans". In: *arXiv preprint arXiv:2508.14006*.

ARK (2026). *ARK: AI-Powered Architectural Design*. URL: https://www.arkdesign.ai (visited on 06/12/2026).

Autodesk (2025). *Autodesk Forma — Environmental Impact Analysis*. URL: https://www.autodesk.com/products/forma (visited on 06/12/2026).

Bagal, Viraj, Rishal Aggarwal, P. K. Vinod, and U. Deva Priyakumar (2022). "MolGPT: Molecular Generation Using a Transformer-Decoder Model". In: *Journal of Chemical Information and Modeling* 62.9, pp. 2064–2076. DOI: 10.1021/acs.jcim.1c00600.

Bao, Fan, Dong-Ming Yan, Niloy J. Mitra, and Peter Wonka (2013). "Generating and Exploring Good Building Layouts". In: *ACM Transactions on Graphics* 32.4, pp. 1–10. DOI: 10.1145/2461912.2461977.

Benedikt, Michael L. (1979). "To Take Hold of Space: Isovists and Isovist Fields". In: *Environment and Planning B: Planning and Design* 6.1, pp. 47–65. DOI: 10.1068/b060047.

Bradley, Ralph Allan and Milton E. Terry (1952). "Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons". In: *Biometrika* 39.3/4, pp. 324–345. DOI: 10.1093/biomet/39.3-4.324.

Brown, Nathan, Marco Fiscato, Marwin H. S. Segler, and Alain C. Vaucher (2019). "GuacaMol: Benchmarking Models for de Novo Molecular Design". In: *Journal of Chemical Information and Modeling* 59.3, pp. 1096–1108. DOI: 10.1021/acs.jcim.8b00839.

Burges, Chris, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender (2005). "Learning to Rank Using Gradient Descent". In: *International Conference on Machine Learning (ICML)*, pp. 89–96.

Chaillou, Stanislas (2019). *ArchiGAN: A Generative Stack for Apartment Building Design*. NVIDIA Developer Technical Blog. Based on the Harvard GSD thesis "AI + Architecture: Towards a New Approach".

Chen, Ting, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton (2020). "A Simple Framework for Contrastive Learning of Visual Representations". In: *International Conference on Machine Learning (ICML)*.

Cho, Kyunghyun, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014). "Learning Phrase Representations Using RNN Encoder–Decoder for Statistical Machine Translation". In: *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. DOI: 10.3115/v1/D14-1179.

Christiano, Paul F., Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei (2017). "Deep Reinforcement Learning from Human Preferences". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Conix.AI (2026). *Conix.AI: AI Layout Generation*. URL: https://www.conix.ai (visited on 06/12/2026).

Evins, Ralph (2013). "A Review of Computational Optimisation Methods Applied to Sustainable Building Design". In: *Renewable and Sustainable Energy Reviews* 22, pp. 230–245. DOI: 10.1016/j.rser.2013.02.004.

Finch 3D (2026). *Finch: Generative Design Tool*. URL: https://www.finch3d.com (visited on 06/12/2026).

Gilmer, Justin, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl (2017). "Neural Message Passing for Quantum Chemistry". In: *International Conference on Machine Learning (ICML)*, pp. 1263–1272.

Gupta, Kamal, Justin Lazarow, Alessandro Achille, Larry Davis, Vijay Mahadevan, and Abhinav Shrivastava (2021). "LayoutTransformer: Layout Generation and Completion with Self-Attention". In: *IEEE/CVF International Conference on Computer Vision (ICCV)*.

Hamilton, William L., Rex Ying, and Jure Leskovec (2017). "Inductive Representation Learning on Large Graphs". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Heschong Mahone Group (2003). *Windows and Offices: A Study of Office Worker Performance and the Indoor Environment*. Tech. rep. California Energy Commission. URL: https://newbuildings.org/resource/windows-and-offices-study-office-worker-performance-and-indoor-environment/ (visited on 06/17/2026).

Heusel, Martin, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter (2017). "GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Hu, Ruizhen, Zeyu Huang, Yuhan Tang, Oliver van Kaick, Hao Zhang, and Hui Huang (2020). "Graph2Plan: Learning Floorplan Generation from Layout Graphs". In: *ACM Transactions on Graphics* 39.4, 118:1–118:14. DOI: 10.1145/3386569.3392391.

Hu, Xiao, Hao Zheng, and Dayi Lai (2024). "Performance Prediction of AI-Generated Architectural Layout Design: Using Daylight Performance of Residential Floorplans as an Example". In: *Proceedings of the Computational Design Symposium (CDAC)*. Shanghai, China.

Hypar, Inc. (2026). *Hypar: Automate Building Design*. URL: https://hypar.io (visited on 06/12/2026).

International Organization for Standardization (2024). *ISO 16032:2024 Acoustics — Measurement of Sound Pressure Level from Service Equipment or Activities in Buildings — Engineering Method*. International Organization for Standardization. URL: https://www.iso.org/standard/83874.html (visited on 06/17/2026).

Jin, Wengong, Regina Barzilay, and Tommi Jaakkola (2018). "Junction Tree Variational Autoencoder for Molecular Graph Generation". In: *International Conference on Machine Learning (ICML)*, pp. 2328–2337.

Kingma, Diederik P. and Max Welling (2014). "Auto-Encoding Variational Bayes". In: *International Conference on Learning Representations (ICLR)*.

Kipf, Thomas N. and Max Welling (2016). "Variational Graph Auto-Encoders". In: *arXiv preprint arXiv:1611.07308*.

— (2017). "Semi-Supervised Classification with Graph Convolutional Networks". In: *International Conference on Learning Representations (ICLR)*.

Ko, Won Hee, Michael G. Kent, Stefano Schiavon, Brendon Levitt, and Giovanni Betti (2022). "A Window View Quality Assessment Framework". In: *LEUKOS: The Journal of the Illuminating Engineering Society* 18.3, pp. 268–293. DOI: 10.1080/15502724.2021.1965889.

Leng, Sicong, Yang Zhou, Mohammed Haroon Dupty, Wee Sun Lee, Sam Joyce, and Wei Lu (2023). "Tell2Design: A Dataset for Language-Guided Floor Plan Generation". In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 14680–14697. DOI: 10.18653/v1/2023.acl-long.820.

Liao, Renjie, Yujia Li, Yang Song, Shenlong Wang, Charlie Nash, William L. Hamilton, David Duvenaud, Raquel Urtasun, and Richard Zemel (2019). "Efficient Graph Generation with Graph Recurrent Attention Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Liu, Jiachen, Yuan Xue, Haomiao Ni, Rui Yu, Zihan Zhou, and Sharon X. Huang (2025). "Computer-Aided Layout Generation for Building Design: A Review". In: *arXiv preprint arXiv:2504.09694*.

Liu, Tie-Yan (2009). "Learning to Rank for Information Retrieval". In: *Foundations and Trends in Information Retrieval* 3.3, pp. 225–331. DOI: 10.1561/1500000016.

Maket (2026). *Maket: Generative Design for Architecture*. URL: https://www.maket.ai (visited on 06/12/2026).

Nagy, Danil, Lorenzo Villaggi, Dale Zhao, and David Benjamin (2017). "Project Discover: An Application of Generative Design for Architectural Space Planning". In: *Proceedings of the Symposium on Simulation for Architecture and Urban Design (SimAUD)*.

Natanian, Jonathan and Thomas Wortmann (2021). "Simplified Evaluation Metrics for Generative Energy-Driven Urban Design: A Morphological Study of Residential Blocks in Tel Aviv". In: *Energy and Buildings* 240, p. 110916. DOI: 10.1016/j.enbuild.2021.110916.

Nauata, Nelson, Kai-Hung Chang, Chin-Yi Cheng, Greg Mori, and Yasutaka Furukawa (2020). "House-GAN: Relational Generative Adversarial Networks for Graph-Constrained House Layout Generation". In: *European Conference on Computer Vision (ECCV)*, pp. 162–177.

Nauata, Nelson, Sepidehsadat Hosseini, Kai-Hung Chang, Hang Chu, Chin-Yi Cheng, and Yasutaka Furukawa (2021). "House-GAN++: Generative Adversarial Layout Refinement Networks". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13632–13641.

Nguyen, Anh-Tuan, Sigrid Reiter, and Philippe Rigo (2014). "A Review on Simulation-Based Optimization Methods Applied to Building Performance Analysis". In: *Applied Energy* 113, pp. 1043–1058. DOI: 10.1016/j.apenergy.2013.08.061.

Oord, Aäron van den, Nal Kalchbrenner, and Koray Kavukcuoglu (2016). "Pixel Recurrent Neural Networks". In: *International Conference on Machine Learning (ICML)*, pp. 1747–1756.

Østergård, Torben, Rasmus L. Jensen, and Steffen E. Maagaard (2016). "Building Simulations Supporting Decision Making in Early Design — A Review". In: *Renewable and Sustainable Energy Reviews* 61, pp. 187–201. DOI: 10.1016/j.rser.2016.03.045.

Polykovskiy, Daniil et al. (2020). "Molecular Sets (MOSES): A Benchmarking Platform for Molecular Generation Models". In: *Frontiers in Pharmacology* 11, p. 565644. DOI: 10.3389/fphar.2020.565644.

Qbiq (2026). *Qbiq: AI Space Planning*. URL: https://www.qbiq.ai (visited on 06/12/2026).

Qin, Sizhong, Ramon Elias Weber, and Xinzheng Lu (2026). "Tokenization Allows Multimodal Large Language Models to Understand, Generate and Edit Architectural Floor Plans (HouseMind)". In: *arXiv preprint arXiv:2603.11640*.

Shabani, Mohammad Amin, Sepidehsadat Hosseini, and Yasutaka Furukawa (2023). "HouseDiffusion: Vector Floorplan Generation via a Diffusion Model with Discrete and Continuous Denoising". In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5466–5475.

Shi, Chence, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang (2020). "GraphAF: A Flow-Based Autoregressive Model for Molecular Graph Generation". In: *International Conference on Learning Representations (ICLR)*.

Shi, Xing and Wenjie Yang (2013). "Performance-Driven Architectural Design and Optimization Technique from a Perspective of Architects". In: *Automation in Construction* 32, pp. 125–135. DOI: 10.1016/j.autcon.2013.01.015.

Shim, Jonghwa, Jaeuk Moon, Hyeonwoo Kim, and Eenjun Hwang (2024). "FloorDiffusion: Diffusion Model-Based Conditional Floorplan Image Generation Method Using Parameter-Efficient Fine-Tuning and Image Inpainting". In: *Journal of Building Engineering* 95, p. 110320. DOI: 10.1016/j.jobe.2024.110320.

Snaptrude (2026). *Snaptrude: Collaborative BIM Modelling*. URL: https://www.snaptrude.com (visited on 06/12/2026).

Sohn, Kihyuk, Honglak Lee, and Xinchen Yan (2015). "Learning Structured Output Representation Using Deep Conditional Generative Models". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Spacio (2026). *Spacio: AI Floor-Plan Generation*. URL: https://www.spacio.ai (visited on 06/12/2026).

Sun, Jiahui, Wenming Wu, Ligang Liu, Weiwei Min, Gaofeng Zhang, and Liping Zheng (2022). "WallPlan: Synthesizing Floorplans by Learning to Generate Wall Graphs". In: *ACM Transactions on Graphics* 41.4, 92:1–92:14. DOI: 10.1145/3528223.3530135.

Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). "Sequence to Sequence Learning with Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

SWAPP (2026). *SWAPP: AI-Driven Architectural Documentation*. URL: https://www.swapp.ai (visited on 06/12/2026).

TestFit, Inc. (2026). *TestFit: Real Estate Feasibility Platform*. URL: https://www.testfit.io (visited on 06/12/2026).

Veličković, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio (2018). "Graph Attention Networks". In: *International Conference on Learning Representations (ICLR)*.

Westermann, Paul and Ralph Evins (2019). "Surrogate Modelling for Sustainable Building Design — A Review". In: *Energy and Buildings* 198, pp. 170–186. DOI: 10.1016/j.enbuild.2019.05.057.

World Health Organization Regional Office for Europe (2009). *Night Noise Guidelines for Europe*. Tech. rep. World Health Organization Regional Office for Europe. URL: https://www.who.int/europe/publications/i/item/9789289041737 (visited on 06/17/2026).

Wortmann, Thomas, Alberto Costa, Giacomo Nannicini, and Thomas Schroepfer (2015). "Advantages of Surrogate Models for Architectural Design Optimization". In: *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 29.4, pp. 471–481. DOI: 10.1017/S0890060415000451.

Wu, Wenming, Xiao-Ming Fu, Rui Tang, Yuhan Wang, Yu-Hao Qi, and Ligang Liu (2019). "Data-Driven Interior Plan Generation for Residential Buildings". In: *ACM Transactions on Graphics* 38.6, 234:1–234:12. DOI: 10.1145/3355089.3356556.

Yang, Kevin et al. (2019). "Analyzing Learned Molecular Representations for Property Prediction". In: *Journal of Chemical Information and Modeling* 59.8, pp. 3370–3388. DOI: 10.1021/acs.jcim.9b00237.

Ye, Ziqi, Sirui Liu, Zhen Tian, Yile Chen, Liang Zheng, and Junming Chen (2025). "Graph-RWGAN: A Method for Generating House Layouts Based on Multi-Relation Graph Attention Mechanism". In: *Buildings* 15.19, p. 3623. DOI: 10.3390/buildings15193623.

Yenew, Azmeraw Bekele, Beakal Gizachew Assefa, and Elefelious Getachew Belay (2024). "HouseGanDi: A Hybrid Approach to Strike a Balance of Sampling Time and Diversity in Floorplan Generation". In: *IEEE Access* 12, pp. 125235–125252. DOI: 10.1109/ACCESS.2024.3451406.

You, Jiaxuan, Bowen Liu, Rex Ying, Vijay Pande, and Jure Leskovec (2018a). "Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

You, Jiaxuan, Rex Ying, Xiang Ren, William L. Hamilton, and Jure Leskovec (2018b). "GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models". In: *International Conference on Machine Learning (ICML)*, pp. 5708–5717.

Zeng, Pengyu, Wen Gao, Jun Yin, Pengjian Xu, and Shuai Lu (2024). "Residential Floor Plans: Multi-Conditional Automatic Generation Using Diffusion Models". In: *Automation in Construction* 162, p. 105374. DOI: 10.1016/j.autcon.2024.105374.

Zhang, Muhan and Yixin Chen (2018). "Link Prediction Based on Graph Neural Networks". In: *Advances in Neural Information Processing Systems (NeurIPS)*.

Zong, Ziyang, Guanying Chen, Zhaohuan Zhan, Fengcheng Yu, and Guang Tan (2024). "HouseTune: Two-Stage Floorplan Generation with LLM Assistance". In: *arXiv preprint arXiv:2411.12279*.

Appendix A

Full Feature Definitions

This appendix gives the exact definition of every input the models of Chapter 3 consume. Section A.1 defines the 30-dimensional per-room feature vector, Section A.2 the two plan-level zone-weight vectors that are concatenated to it, and Section A.3 the conditioning (budget) context supplied to the zone decoder. The names in the `typewriter` font are the literal field identifiers used in the dataset and training code, and the dimension indices give the order in which the values appear in the feature tensor.

A.1 Per-Room Feature Vector

Each room is represented by a 30-dimensional vector: 21 numerical features (dimensions 1–21) followed by a 9-way one-hot encoding of the room type (dimensions 22–30). Plans are padded to a maximum of 20 rooms; padding rows are all-zero and masked in every loss and attention computation.

The room-type one-hot encoding occupies dimensions 22–30 in the fixed order of Table A.2. Exactly one of these dimensions is 1.

Normalisation. The 21 numerical features are standardised to zero mean and unit variance using a single scaler fitted on a large random sample of training rooms; the nine one-hot columns are left as raw 0/1 indicators and are not scaled. The same fitted scaler is reused at inference, so generation sees features on the same scale as training.

Table A.1: The 21 numerical room features (dimensions 1–21), in tensor order.

Dim	Identifier	Definition
1	`area`	Room floor area in plan units (raw polygon area).
2	`width_norm`	Bounding-box width, normalised by the plan extent.
3	`height_norm`	Bounding-box height, normalised by the plan extent.
4	`centroid_x_norm`	Room-centroid x coordinate, normalised to $[0, 1]$ over the plan.
5	`centroid_y_norm`	Room-centroid y coordinate, normalised to $[0, 1]$ over the plan.
6	`n_exposed_facades`	Number of cardinal facades on the plan boundary, 0–4.
7	`wwr`	Overall window-to-wall ratio (window length over exposed wall length), capped at 0.95.
8	`wwr_n`	WWR on the north facade.
9	`wwr_s`	WWR on the south facade.
10	`wwr_e`	WWR on the east facade.
11	`wwr_w`	WWR on the west facade.
12	`acoustic_wwr`	Acoustically-effective opening ratio, $\min(\texttt{wwr} + 5.01\, e,\ 2.0)$, where e is the door-opening ratio; doors transmit sound, so they augment the effective aperture.
13	`acoustic_wwr_n`	Acoustic opening ratio, north.
14	`acoustic_wwr_s`	Acoustic opening ratio, south.
15	`acoustic_wwr_e`	Acoustic opening ratio, east.
16	`acoustic_wwr_w`	Acoustic opening ratio, west.
17	`area_fraction`	`area` divided by the total interior area of the plan (the shares sum to 1 across rooms).
18	`qv_potential`	View potential: $\max_z\, v(z)\, \texttt{wwr}$ over the nine zones — the best quality-view score the room could attain in any zone.
19	`indoor_spl`	Baseline estimated indoor sound-pressure level (dB). The zone-dependent value used for scoring is recomputed per placement (Equation 3.11); the padding/fallback value is 10 dB.
20	`acoustic_score`	$\mathrm{clip}((\theta_t - \texttt{indoor\_spl})/(\theta_t - 10),\ 0,\ 1)$ for the three AQI room types ($\theta_{\text{bedroom}} = 35$, $\theta_{\text{living}} = \theta_{\text{kitchen}} = 40$ dB); 0 for all other types.
21	`has_balcony_access`	Binary flag: 1 if the room adjoins a balcony (read from the plan graph links), else 0.

Table A.2: Room-type one-hot encoding (dimensions 22–30).

Dim	22	23	24	25	26	27	28	29	30
Type	living	kitchen	bedroom	bathroom	dining	study	office	laundry	other

A.2 Plan-Level Zone-Weight Vectors

Before encoding, each room vector is concatenated with two plan-level vectors of length nine — the view zone weights and the acoustic zone weights — giving the $30 + 9 + 9 = 48$-dimensional encoder input of Equation 3.14. Both are defined over the 3×3 zoning, whose exposed facades are listed in Table A.3.

Table A.3: The nine zones (row-major) and the cardinal facades each one exposes. Zone 5 is the sheltered interior.

Zone	Position	Exposed facades	# facades
1	NW	N, W	2
2	N	N	1
3	NE	N, E	2
4	W	W	1
5	C	—	0
6	E	E	1
7	SW	S, W	2
8	S	S	1
9	SE	S, E	2

View zone weights $v(z)$. Each cardinal direction is assigned a categorical view quality $c(\cdot) \in [0, 1]$ — vegetation 1.00, sky 0.90, urban 0.70, balcony 0.40, no view 0.00 — and a zone's view weight is the mean of c over the facades it exposes (Equation 3.6); the interior zone has weight 0.

Acoustic zone weights $\hat{a}(z)$. Each zone's outdoor noise energy is computed from the per-direction outdoor levels and then range-normalised across the nine zones to lie in $[0, 1]$, so the quietest (interior) zone maps to 0 and the noisiest to 1. The per-direction outdoor ranges are asymmetric: north 62–78, south 65–80, east 63–79, and west 60–77 dB.

A.3 Conditioning (Budget) Context

The zone decoder receives, at every step, the 31-dimensional conditioning vector of Equation 3.22, summarised in Table A.4. The first three entries are the user-supplied targets; the remainder are structural progress signals and the plan-level zone weights.

Table A.4: The 31-dimensional conditioning (budget) context.

Dims	Component	Definition
1–3	Target triple	Requested QVCI, AQI, and spread (each in $[0, 1]$).
4	Rooms remaining	Fraction of rooms not yet placed, $\rho_t = (N - t)/N$.
5–13	Zone-usage counts	Running normalised count of rooms already assigned to each of the nine zones.
14–22	View zone weights	The nine $v(z)$ of Section A.2.
23–31	Acoustic zone weights	The nine $\hat{a}(z)$ of Section A.2.

Appendix B

Arrangement-Type Generation Details

The zone model is trained on a contrastive dataset (Section 3.3.1) in which each reference plan is paired with many alternative zone assignments that span good and bad environmental performance, so the model learns what separates a strong arrangement from a weak one rather than memorising a single ground truth. This appendix documents exactly how those alternatives are generated.

B.1 Corpus Scale

Each of the 16,734 source plans is expanded by the eight symmetry transforms of the dihedral group D_4 (identity, three rotations, two mirror flips, and two diagonal reflections), and within each transformed plan up to 21 arrangements are generated across the eleven families of Table B.1. This gives the full corpus of

$$16{,}734 \times 8 \times 21 = 2{,}811{,}312 \text{ records.}$$

Within each family the generator keeps a set of the zone tuples it has already emitted and discards exact duplicates, so the stochastic families (perturbation, constrained-random, random) never contribute repeated arrangements for the same plan.

Table B.1: The eleven arrangement families, their multiplicity per plan–transform cell, and the training signal each provides.

Family	Count	Training signal
`real`	1	The plan's actual geometric zone assignment — the positive reference.
`perturbation`	3	Near-misses: `real` with one or two random room-pair swaps.
`greedy_best`	1	Upper anchor on the combined view+acoustic objective.
`greedy_worst`	1	Lower anchor on the combined objective.
`acoustic_greedy_best`	1	Pure acoustic signal: quietest zones first.
`acoustic_greedy_worst`	1	Pure acoustic signal: noisiest zones first.
`all_in_best_zone`	1	Degenerate clustering: every room in the single best-view zone.
`all_in_two_zones`	1	Degenerate clustering across the top two view zones.
`type_clustered`	1	Every room type confined to its own zone.
`constrained_random`	5	Type-diverse random baselines.
`random`	5	Unconstrained random baselines.
Total	**21**	

B.2 The Eleven Arrangement Families

B.3 Generation Rules

Combined objective. The greedy families score each zone z by a weighted blend of its view weight $v(z)$ and its acoustic quality $1-\hat{a}(z)$ (the complement of the acoustic zone weight, so the sheltered interior scores highest):

$$s(z) = \alpha\, v(z) + (1-\alpha)\,(1-\hat{a}(z)), \qquad \alpha = 0.6. \tag{B.1}$$

The view weight $\alpha = 0.6$ reflects that quality view is the primary objective. Rooms are placed largest-area-first into zones taken in ranked order.

Greedy families. `greedy_best` walks the zones from highest to lowest $s(z)$; `greedy_worst` walks them in the reverse order. The `acoustic_greedy` pair uses the same procedure but ranks zones by acoustic quality $1-\hat{a}(z)$ alone, ignoring view. Supplying the acoustic ranking separately from the combined ranking gives the model an acoustic conditioning signal that is decoupled from view, which matters because the highest-view zones are also the noisiest; without the decoupled pair the model would only ever see view and acoustics moving together.

Perturbation. Starting from `real`, one or two random pairs of rooms have their zones swapped. These sit close to the reference in arrangement space and teach the model the fine boundary around a good layout.

Degenerate clustering. `all_in_best_zone` places every room in the single highest-view zone; `all_in_two_zones` alternates rooms between the top two view zones; and `type_clustered` gives each room type its own zone,

with zones assigned in descending view-weight order. These are structurally implausible but performance-instructive: they show the model that concentration is penalised by the spread objective even when it chases view.

Random baselines. `constrained_random` draws a zone per room uniformly while discouraging a room type from reusing a zone it already occupies; `random` draws each room's zone uniformly with no constraint. Together they populate the low-quality region of the performance space and prevent the model from treating any plausible arrangement as automatically good.

Appendix C

Hyperparameter Tables

This appendix collects the architecture and training hyperparameters for the three trained models of the pipeline (Stages 1–3) and the shared physical constants used to compute the performance metrics. The values correspond to the configuration that produced the results of Chapter 4.

C.1 Stage 1 — Area Distribution Model

The area model is a conditional logistic-normal density network: a feed-forward encoder maps the conditioning inputs to a latent vector, from which a mean head and a log-standard-deviation head parameterise a distribution over the room-area shares on the simplex (Section 3.2.2).

Table C.1: Stage 1 (area distribution) hyperparameters.

Setting	**Value**
Architecture	Feed-forward MLP, logistic-normal output
Hidden width	128
Latent width	16
Output heads	mean and log-standard-deviation (one per area share)
Log-σ clamp	$[-5, 1]$
Dropout	0.10
Monte-Carlo samples (loss)	8
Optimiser	AdamW
Learning rate	1×10^{-3}
Weight decay	1×10^{-5}
Batch size	256
Epochs	150
Gradient-norm clip	5.0
Train/validation split	90/10
Random seed	42

C.2 Stage 2 — Zone Assignment Model

The zone model is the core conditional Graph-to-Sequence network: a multi-head graph-attention encoder over the room graph feeds an autoregressive GRU decoder that emits one zone per room (Section 3.3.5 and following). The 48-dimensional encoder input (the 30-dimensional room vector plus the two nine-dimensional zone-weight vectors) is projected to the GAT width.

Table C.2: Stage 2 (zone assignment) hyperparameters.

Setting	**Value**
Encoder	Multi-head graph attention (GAT)
layers / heads / width	2 / 4 / 128
input projection	$48 \to 128$
Decoder	Autoregressive GRU
layers / width	2 / 256
Classifier head	$256 \to 128 \to 9$
Per-step decoder input	$128 + 31 + 9 = 168$
Budget context width	31
Max rooms (padding)	20
Dropout	0.10
Parameters	$\approx$ 796,000
Optimiser	Adam
Learning rate	5×10^{-4}
LR schedule	StepLR, halved every 30 epochs ($\gamma = 0.5$)
Batch size	128
Epochs	100
Gradient-norm clip	5.0
Target-conditioning rule	70/30 (Section 3.3.8)
Random seed	42

C.3 Stage 3 — Connectivity Model

The connectivity model is a decoder-only autoregressive predictor over the lower-triangular sequence of possible room pairs. Because the inputs are purely categorical, embedding lookups replace the graph encoder (Section 3.4.4): the two endpoint type embeddings and the previous-edge embedding are concatenated and fed to a single GRU whose final hidden state serves as the graph embedding.

C.4 Shared Physical and Metric Constants

Table C.4 lists the constants used to compute QVCI and AQI and to drive the contrastive arrangement generation. They are shared across the dataset builder, the training scripts, and the generators.

Table C.3: Stage 3 (connectivity) hyperparameters.

Setting	**Value**
Architecture	Decoder-only autoregressive GRU over edge decisions
Room-type embedding	128 (per endpoint)
Previous-edge embedding	16
GRU input width	$128 \times 2 + 16 = 272$
GRU	single layer, width 512
Output head	linear $\rightarrow$ edge logit (sigmoid)
Optimiser	AdamW
Learning rate	1×10^{-3}
Weight decay	1×10^{-4}
Batch size	16
Epochs	200
Gradient-norm clip	1.0
Random seed	42

Table C.4: Physical and metric constants.

Constant	**Value**
Sound-reduction index, wall (STC)	45
Sound-reduction index, window	32
Sound-reduction index, door	25
Outdoor SPL range, north	62–78 dB
Outdoor SPL range, south	65–80 dB
Outdoor SPL range, east	63–79 dB
Outdoor SPL range, west	60–77 dB
AQI comfort thresholds	bedroom 35, living 40, kitchen 40 dB
AQI lower reference $L_{\min}$	10 dB
Quality-view indicator threshold τ_{QV}	0.15
Door-opening (EDR) weight	5.01
Ceiling height (plan units)	70
Sabine absorption coefficient α_{room}	0.15
Combined-objective view weight α	0.6

Appendix D

Additional Generated Layouts

This appendix collects the complete target sweep summarized in Chapter 4. Each contact sheet shows the top nine of twelve candidate schematic layouts generated for the representative ten-room plan under one performance-target setting. The panel headers report each candidate's achieved QVCI (Q) and AQI (A). The compact scatter view of the same sweep is shown in Figure 4.5 in the main text.

The appendix includes four sets of additional outputs:

- **Acoustic-target sweep with view target fixed at** 0.50**:** the view target is held constant while the acoustic target is increased across the tested range.
- **View-target sweep with acoustic target fixed at** 0.50**:** the acoustic target is held constant while the view target is increased across the tested range.
- **Connectivity prediction examples:** predicted connectivity outputs are shown for plans with 5 rooms, 7 rooms, and 13 rooms.
- **Area-distribution prediction:** the predicted area distribution is shown for the representative 10-room case.

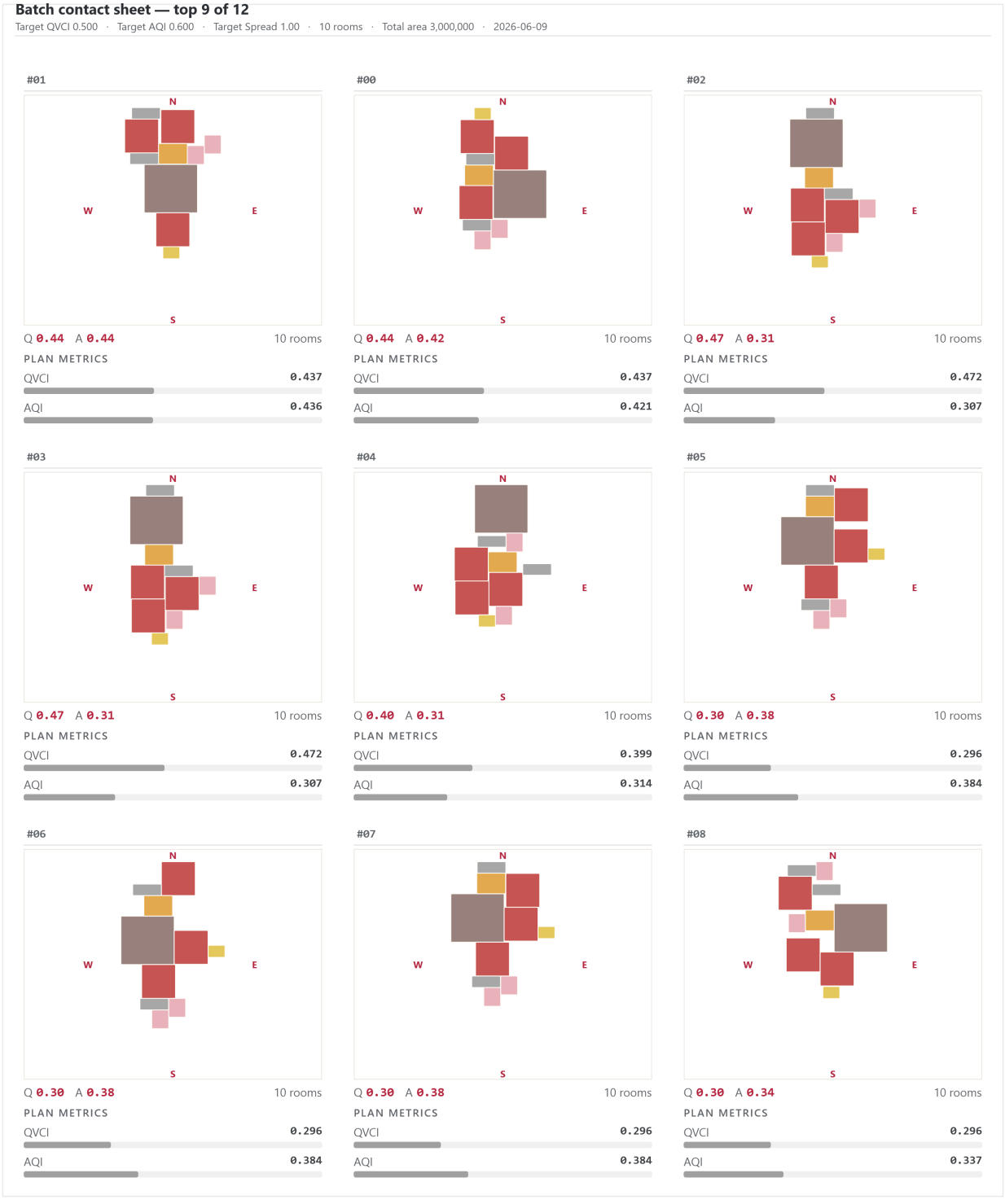


Figure D.1: Generated layout candidates for target QVCI = 0.50, AQI = 0.60.

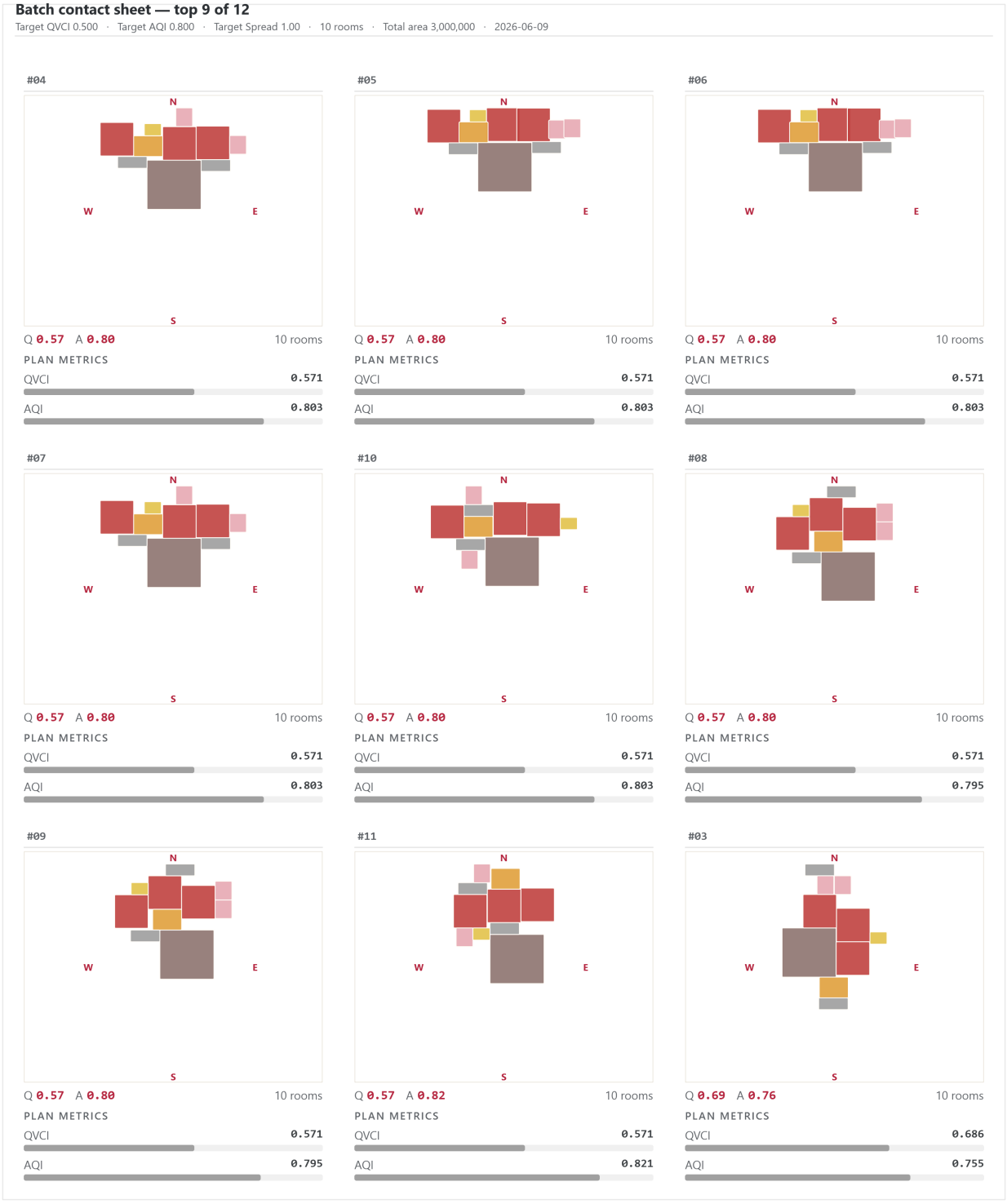


Figure D.2: Generated layout candidates for target QVCI = 0.50, AQI = 0.80.

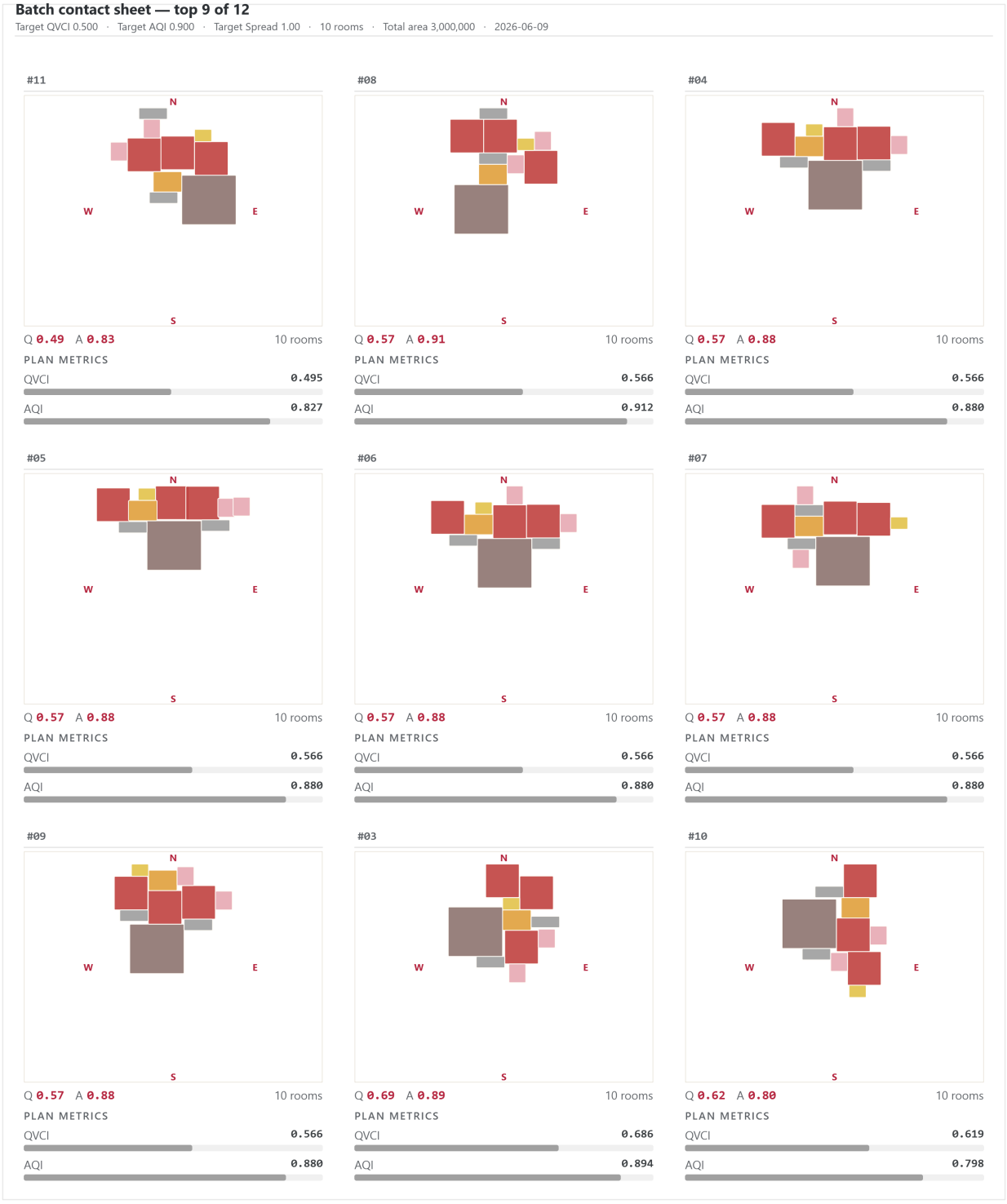


Figure D.3: Generated layout candidates for target QVCI = 0.50, AQI = 0.90.

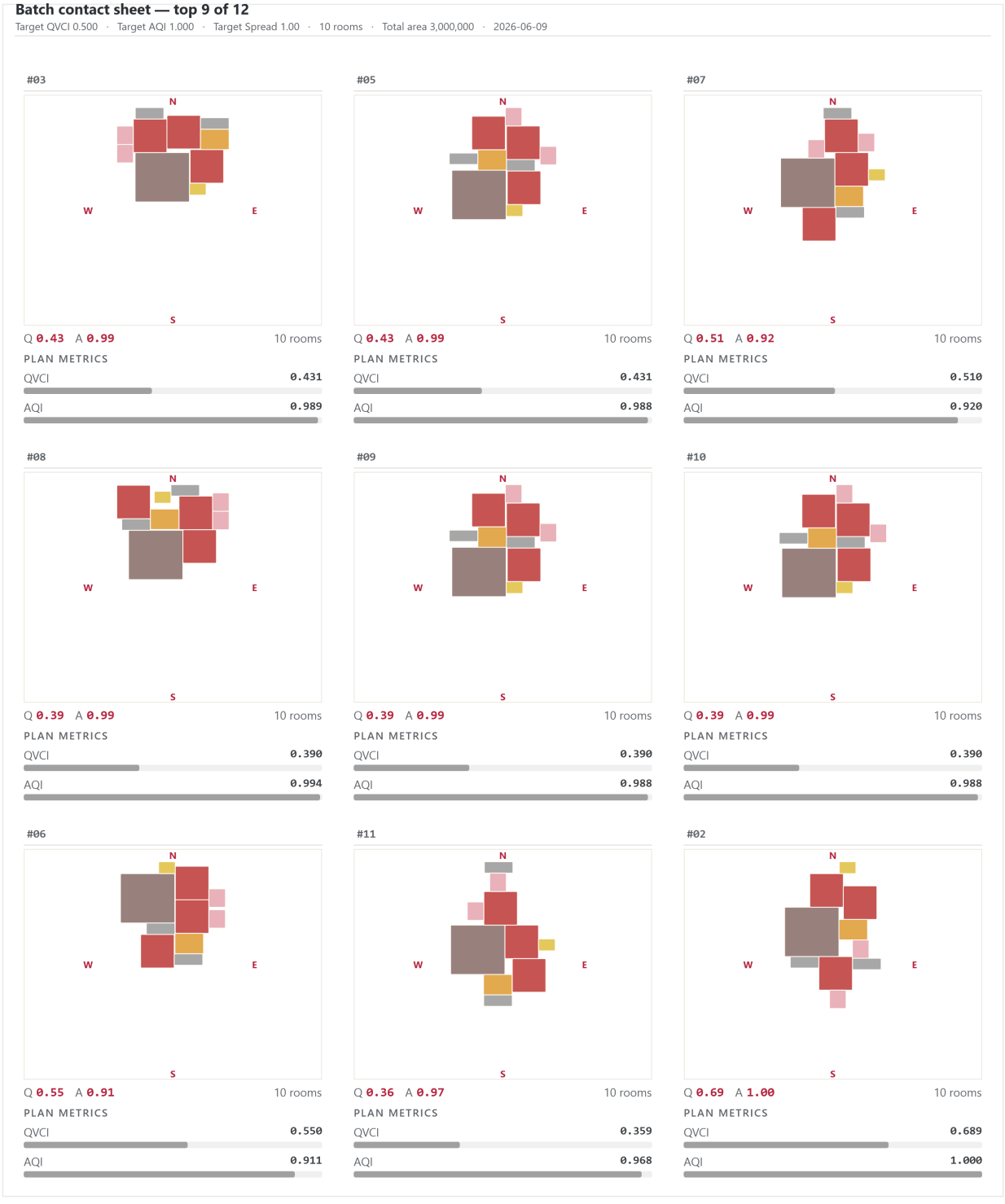


Figure D.4: Generated layout candidates for target QVCI = 0.50, AQI = 1.00.

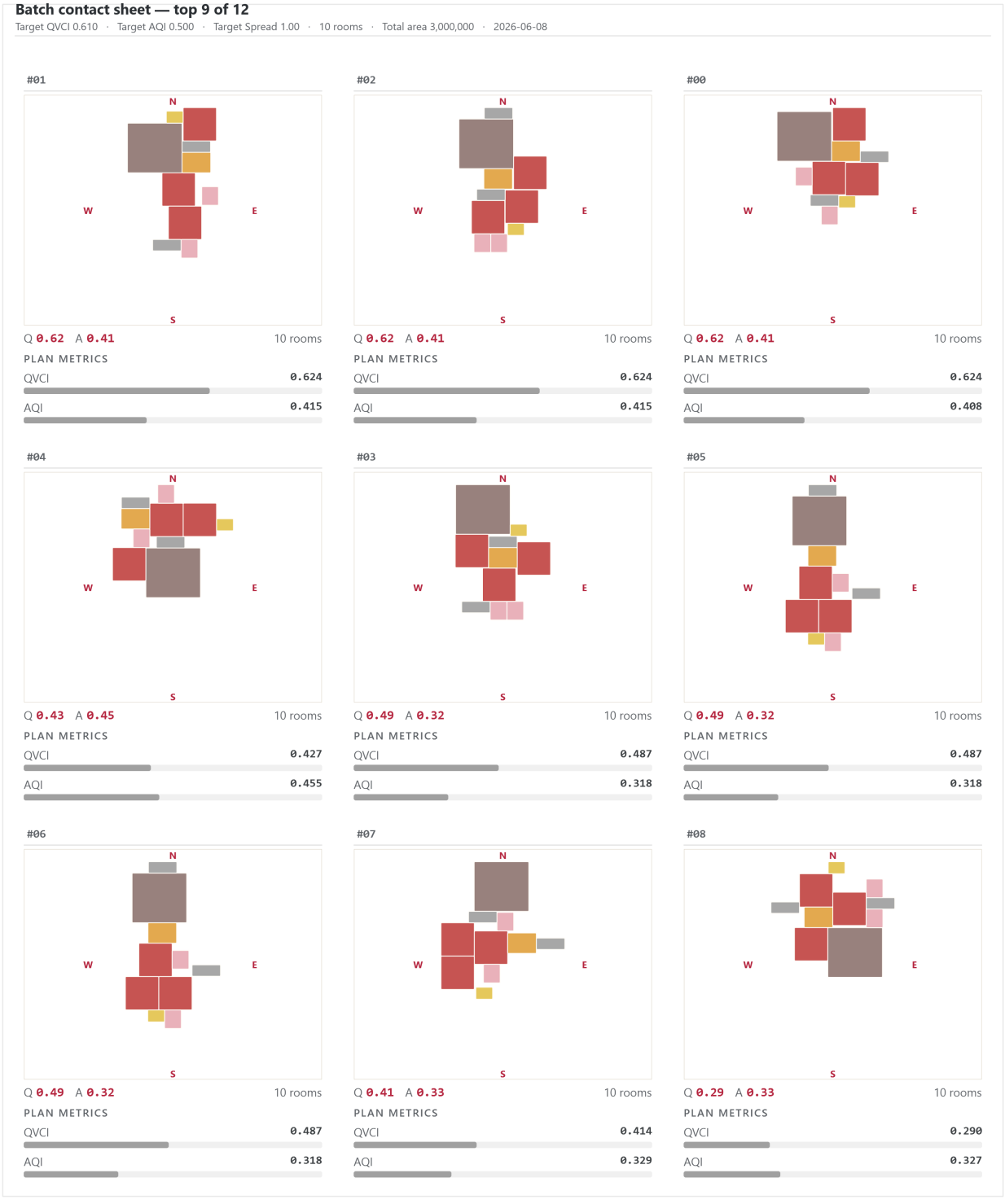


Figure D.5: Generated layout candidates for target QVCI = 0.60, AQI = 0.50.

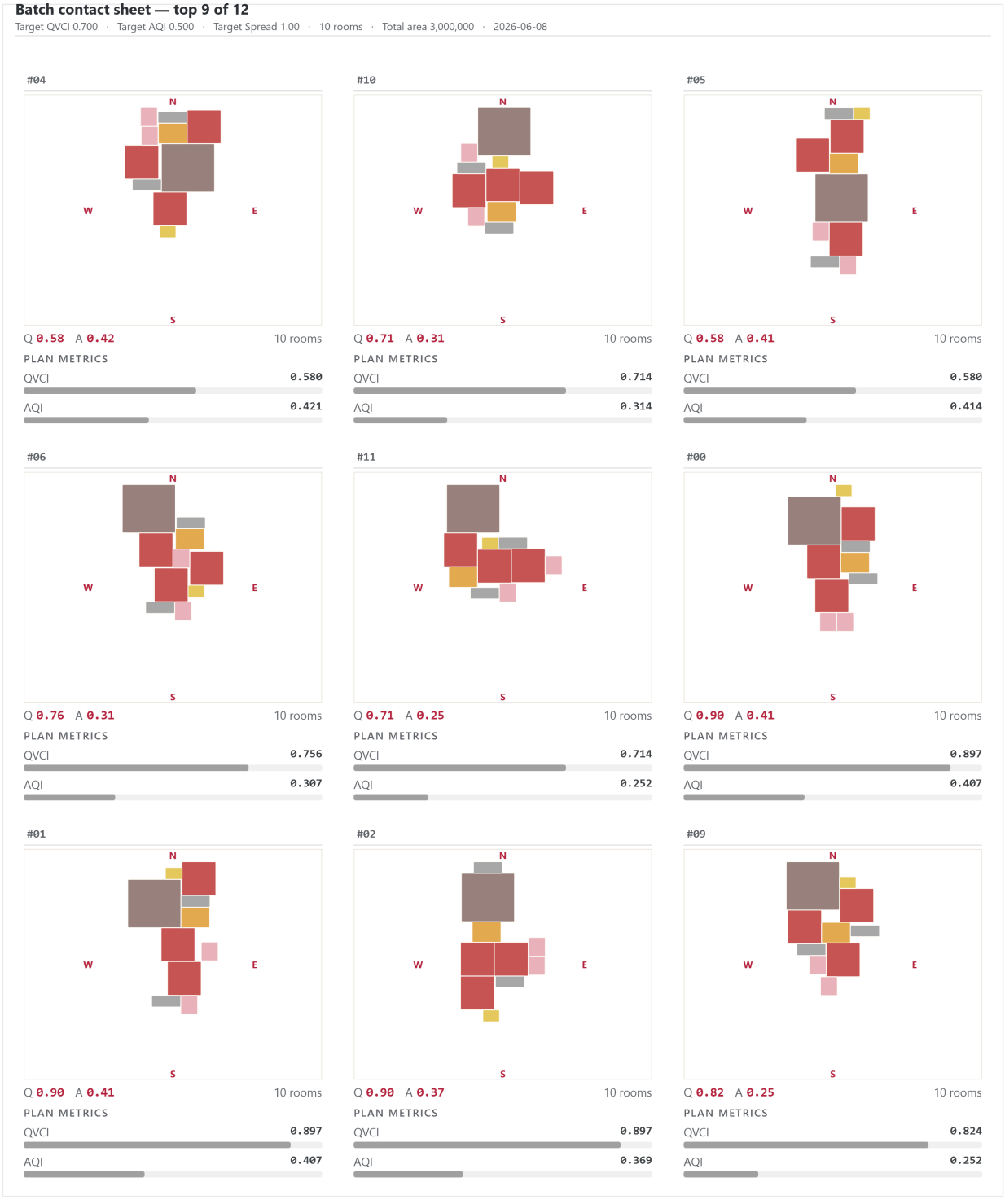


Figure D.6: Generated layout candidates for target QVCI = 0.70, AQI = 0.50.

Figure D.7: Generated layout candidates for target QVCI = 0.80, AQI = 0.50.

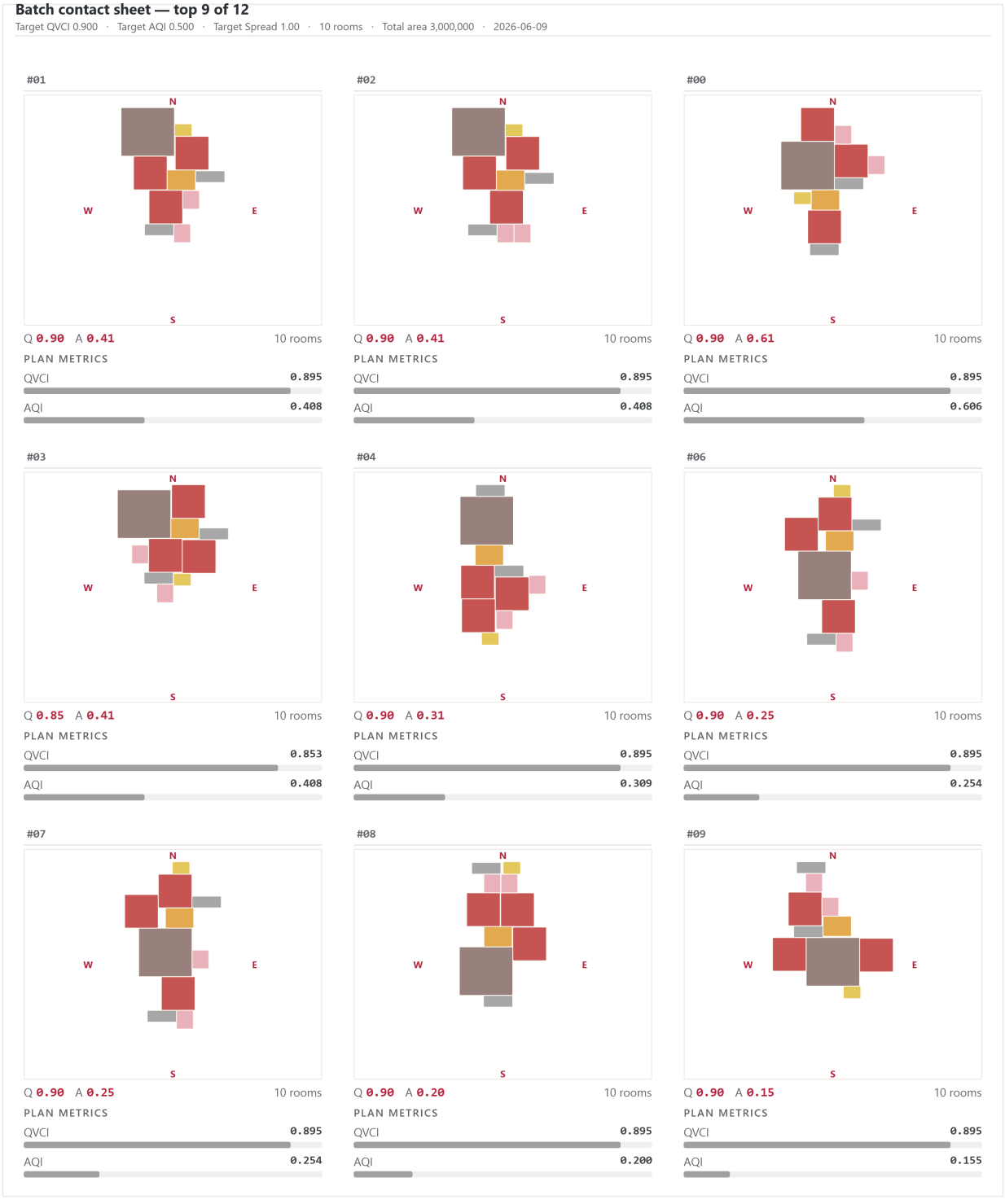


Figure D.8: Generated layout candidates for target QVCI = 0.90, AQI = 0.50.

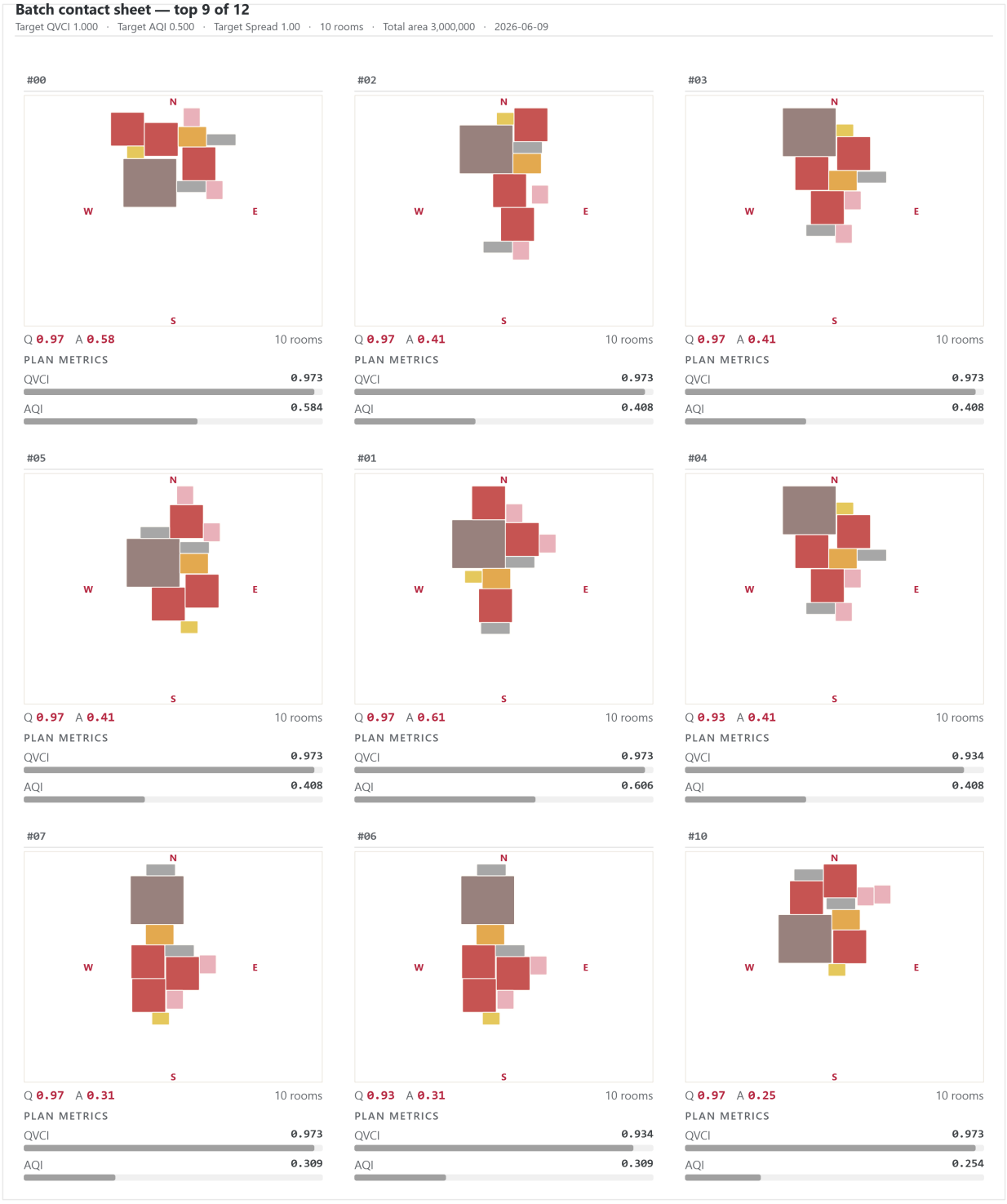


Figure D.9: Generated layout candidates for target QVCI = 1.00, AQI = 0.50.

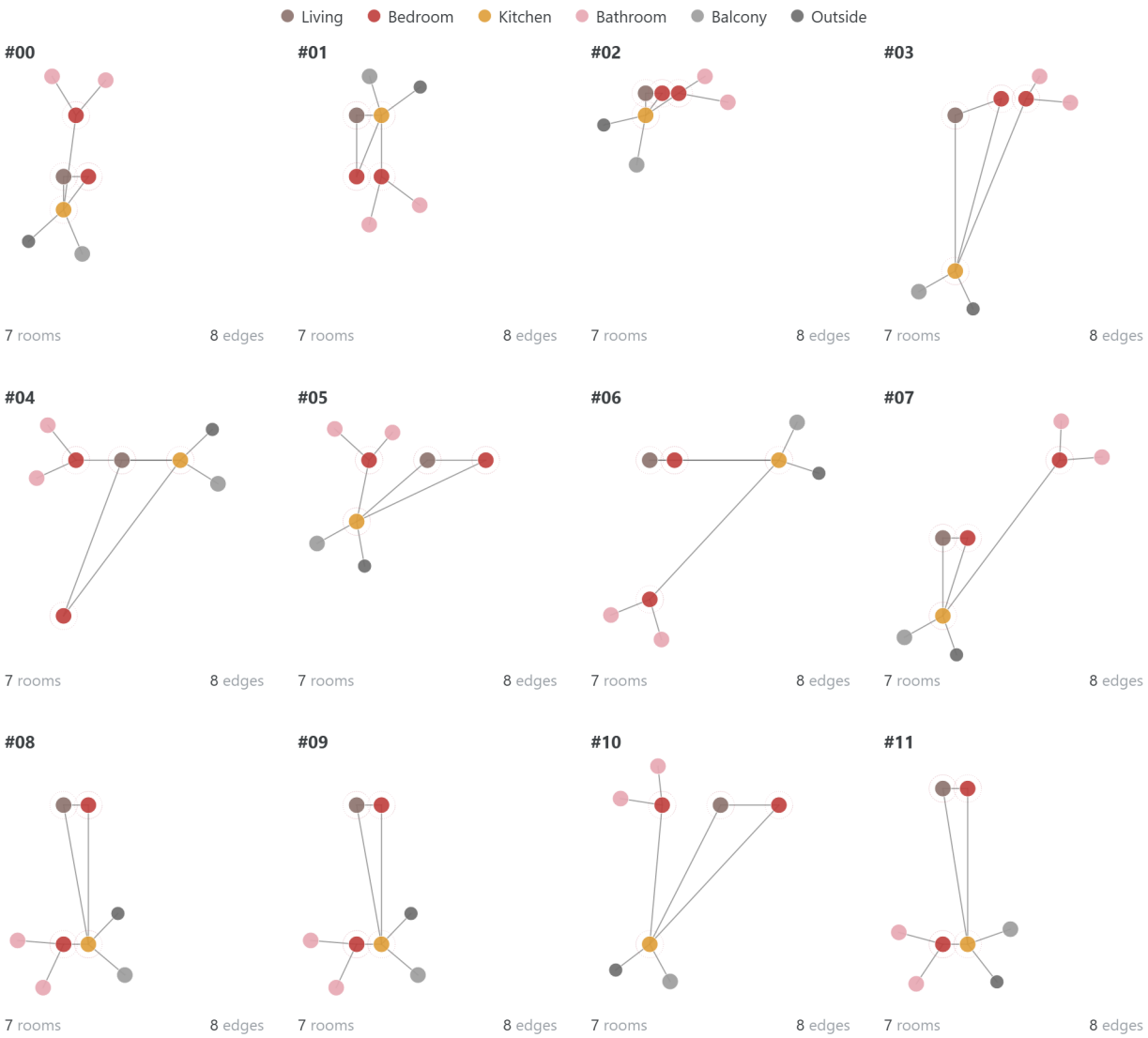


Figure D.10: A second batch of generated room-connectivity graphs (companion to Figure 4.1). Node colour encodes room type; each plan has seven rooms joined by eight edges.

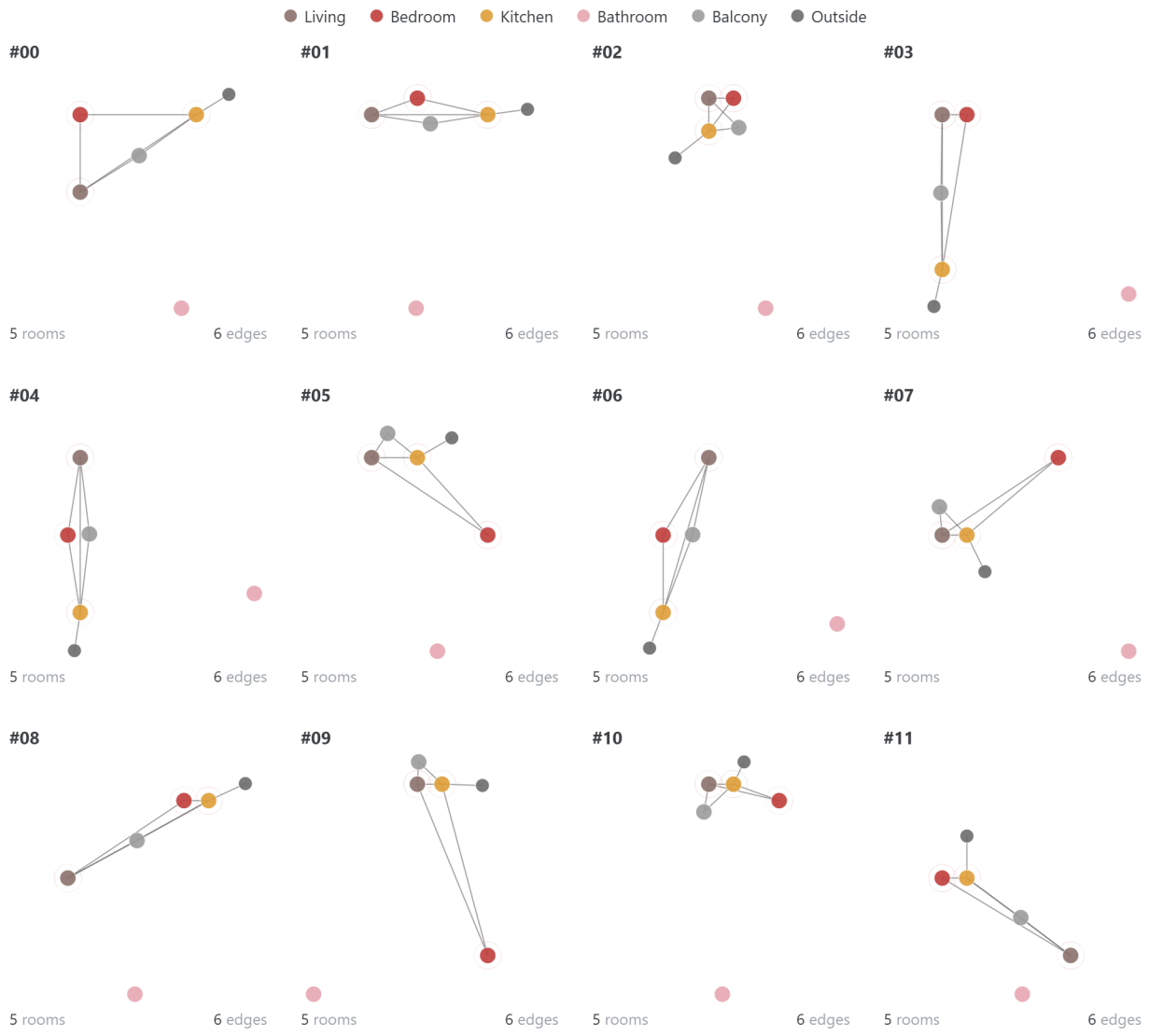


Figure D.11: A third batch of generated room-connectivity graphs (companion to Figure 4.1). Node colour encodes room type; each plan has five rooms joined by six edges.

Figure D.12: A fourth batch of generated room-connectivity graphs (companion to Figure 4.1). Node colour encodes room type; each plan has thirteen rooms joined by sixteen edges.

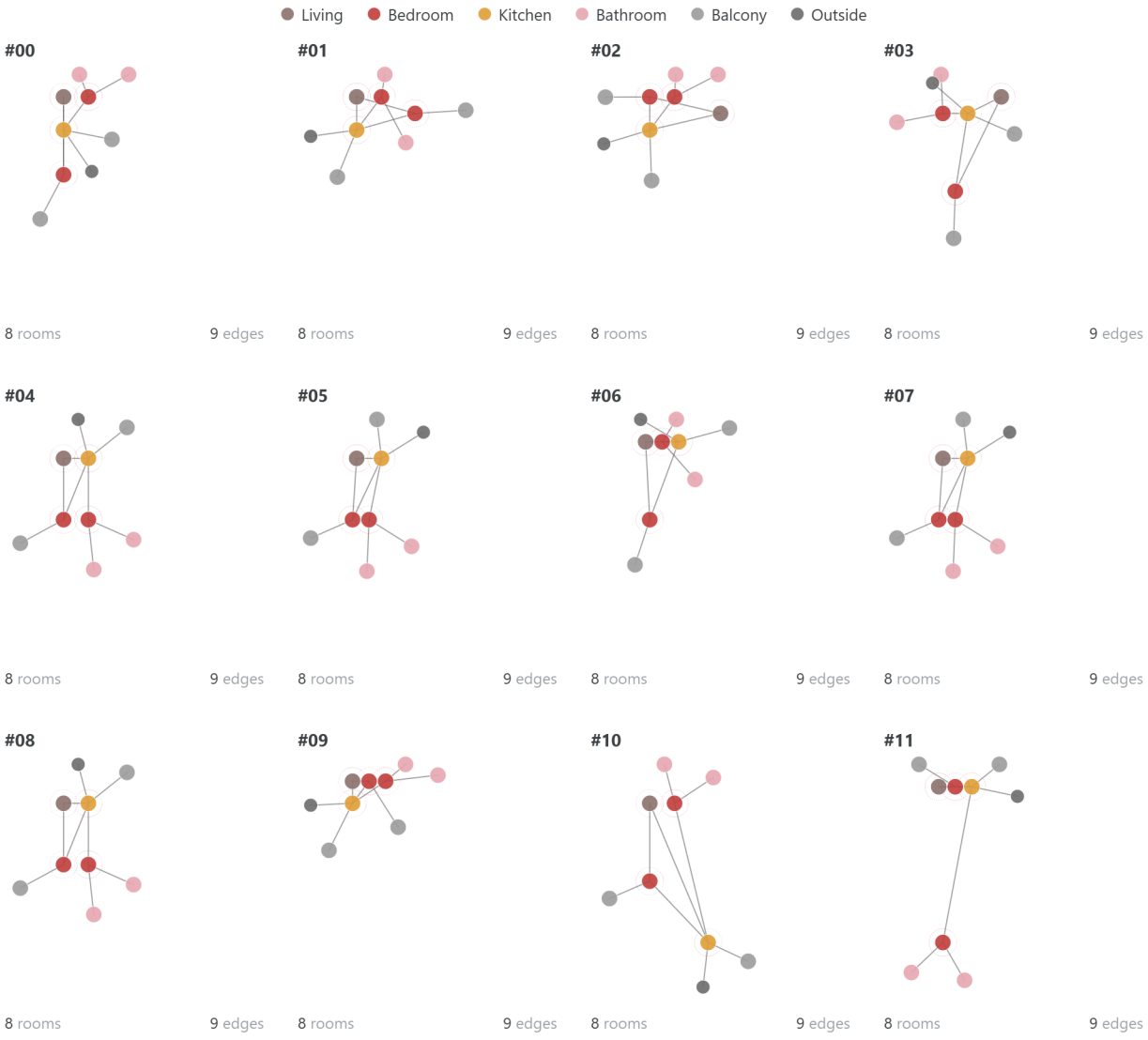


Figure D.13: A fifth batch of generated room-connectivity graphs (companion to Figure 4.1). Node colour encodes room type; each plan has ten rooms joined by ten edges.

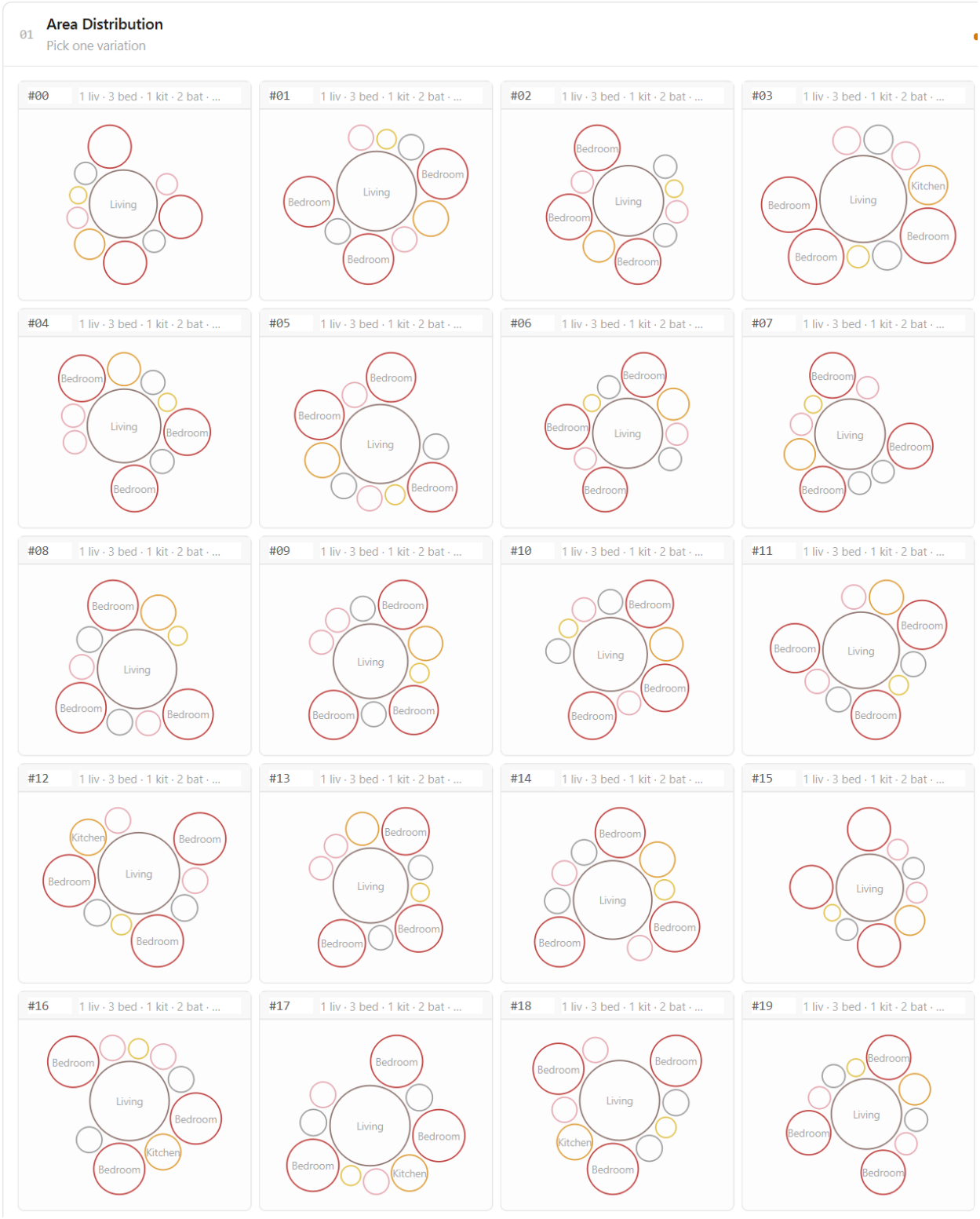


Figure D.14: Twenty candidate area distributions from the Stage 1 area model (Section 3.2) for one fixed room program (one living room, three bedrooms, one kitchen, two bathrooms, and service rooms). Each card (#00–#19) is an independent logistic-normal sample (Equation (3.2)), so all share the program but differ in how the total floor area is split. Every room is a circle whose area is proportional to its predicted floor area, coloured by type — bedrooms red, living taupe, kitchens amber, bathrooms pink, other rooms grey — with the largest room anchoring the centre. The spread illustrates the diversity of the sampled distributions; the user selects one to carry forward to zone assignment (Stage 3.3).

Appendix E

Reproducibility (Code and Data Availability)

This appendix records the computing environment, the pipeline of scripts, and the data provenance needed to reproduce the results of this thesis.

E.1 Computing Environment

All models were trained and evaluated on a single consumer laptop, confirming that the pipeline is lightweight enough to run without specialised hardware (Table E.1).

Table E.1: Computing environment.

Component	**Specification**
Machine	ASUS ROG Strix G513IM laptop
GPU	NVIDIA GeForce RTX 3060 Laptop (6 GB VRAM)
System memory	16 GB RAM
Operating system	Windows 11 (PowerShell)
Python	3.10 or newer
Deep-learning framework	PyTorch (CUDA 12.x build)
Key libraries	NumPy, scikit-learn, Matplotlib, ijson (streaming the raw JSON), Plotly (3-D viewer), PySide6 (desktop application), FastAPI and JavaScript (web app)
Random seed	42 throughout (data generation, conversion, and all three training runs)

E.2 Pipeline

The end-to-end pipeline runs as an ordered sequence of scripts, each consuming the output of the previous stage (Table E.2). Fixing the seed at every stage makes the dataset construction, the train/validation split,

and the training trajectory deterministic on the same machine.

Table E.2: Reproduction pipeline, in execution order.

Script	Role
`build_contrastive_dataset_v6.py`	Generates the contrastive dataset (≈ 14 GB JSON) from the screened plans.
`audit_v12_streaming.py`	Streams the dataset and verifies plan count, record count, and balance checks.
`convert_to_npz.py`	Fits the feature scaler on a 200,000-room sample, then writes the standardised memory-mapped tensors used for training.
`Area_distribution_train_code.py`	Trains the Stage 1 area model.
`train_v12b.py`	Trains the Stage 2 zone model.
`Connectivity_train_code.py`	Trains the Stage 3 connectivity model.
`visualise_embeddings_v12.py`	Produces the latent-space and evaluation visualisations.

Checkpoint loading note. The training checkpoints are saved as full objects rather than weight dictionaries. On PyTorch 2.6 and newer, where `torch.load` defaults to weights-only loading, the checkpoints must be loaded with `weights_only=False` (the files are the author's own and are trusted).

E.3 Code and Data Availability

Source code. The full pipeline — dataset construction, the three training scripts, the generators, the desktop application and web app — is available at https://github.com/ShafiArch/Layout-Generator_GAT_GRU_Autoregressive, released under the MIT licence.

Trained models. The trained checkpoints for the three stages are archived at https://github.com/ShafiArch/Layout-Generator_GAT_GRU_Autoregressive.

Dataset. The contrastive dataset is derived from the ResPlan residential floor-plan corpus (Abouagour and Garyfallidis 2025), which is publicly released by its authors. The derived contrastive dataset (16,734 plans; 2,811,312 records) and the construction scripts that reproduce it from ResPlan are available at https://github.com/ShafiArch/Layout-Generator_GAT_GRU_Autoregressive. Because the full dataset is large (≈ 14 GB), the construction scripts and the smaller processed tensors are provided so the dataset can be regenerated rather than downloaded in full.