# SmartPlanAI: Floorplan Generation Using instruct-Pix2Pix

Anindya Anil Sur, Anant Khurana†, Anshu Bhakat,
Mohd. Mohsin Ali and Umesh Gupta

*Corresponding author(s). E-mail(s): suranindya629@gmail.com;
anant200414@gmail.com; anshubhakatt@gmail.com;
Contributing authors: mohdmohsin066gmail.com; umeshguptagmail.com;
†These authors contributed equally to this work.

**Abstract**

This project explores the application of deep learning techniques specifically in the field of architectural design. N model, to automate the generation of architectural floorplans from input footprints. We begin by reviewing two prominent developments in the field: ArchiGAN and the pix2pix model, highlighting their unique approaches and challenges and their significant potential in architectural design applications. Leveraging a dataset comprising of over 12000 samples, we develop a robust training pipeline using TensorFlow and Keras libraries. Our methodology incorporates a U-Net-based generator and PatchGAN discriminator, optimized with custom loss functions and Adam optimizers. Through iterative refinement and experimentation, we aimed to generate creative floorplans that are cohesive and within the bounds of the given footprint. By harnessing the power of Conditional Generative Adversarial Networks (C-GANs), this research aims to spotlight architectural design using artificial intelligence and machine learning methods, empowering architects with innovative tools for efficient and creative floorplan generation.

**Keywords:** Conditional Generative Adversarial Networks, C-GAN, Convolutional Neural Networks, CNN, ArchiGAN, Pix2Pix, instruct-Pix2Pix, deep learning, Tensorflow, Generative Adversarial Networks, GAN, floorplan, architectural design

# 1 Introduction

In recent years, Generative Adversarial Networks (GANs)[10] have emerged as powerful tools for generating realistic and high-fidelity images across various domains, including computer vision, art, and architecture. Among the plethora of applications in several industries, architectural design stands out as a particularly promising field where GANs have demonstrated their potential to revolutionize traditional design processes. In this paper, we explore a GAN based approach called *Pix2Pix* [1]. We then explore the efficacy of applying an instruction-based Pix2Pix model to generate floorplans given a footprint area as the input.

Architectural design, while being one of the more creative endeavours, still requires ample technical expertise. The creation of novel designs are often hampered by physical or logical constraints that can hinder efficiency and innovation. Traditionally, architects rely on manual drafting and iterative design processes to explore different layout options for refining their designs. However, these methods are often time-consuming, labour-intensive, and limited in the designer's creative proficiency to explore a range of design alternatives. Furthermore, considering spatial efficiency, integrating various design constraints and client preferences can add to the complexity of the designs[2]. A potential solution to these issues is the utilization of generative models like Generative Adversarial Networks (GANs), which can automatically generate multiple design options based on specific criteria and constraints, expediting the design process and fostering creative exploration. Techniques like reinforcement learning can be employed to optimize designs for aesthetic or regional appeal while adhering to functional requirements. Additionally, Convolutional Neural Networks (CNNs)[11] can be trained to automatically identify potential code violations, ensuring adherence to building regulations. By analyzing large datasets of existing floor plans, deep learning techniques can reveal valuable patterns and insights that inform design decisions, leading to more data-driven and objective design solutions.

The advent of GANs has opened up new avenues for automating and augmenting the architectural design process. By leveraging deep learning techniques, Convolutional Neural Networks (CNNs)[12] and GANs can learn from large datasets of existing building layouts and generate novel designs that exhibit similar characteristics. This capability holds tremendous promise for accelerating the design iteration cycle, facilitating creativity, providing more options for homeowners and ultimately enhancing the quality of architectural outputs.

*ArchiGAN*, developed by Stanislas Chaillou[3], represents a cutting-edge approach to architectural design using GANs[4]. Its unique generative stack integrates programmatic, generative, and layout refinement model components to enable architects and interior designers to generate diverse and realistic apartment building layouts tailored to specific requirements and constraints. Through the interplay of the programmatic model, which encodes the strict design rules and constraints, and the generative model, which produces initial layout proposals, ArchiGAN facilitates the exploration of a vast design space while maintaining coherence and feasibility.[4]

In parallel, *Pix2Pix* has gained prominence as a versatile image-to-image translation framework that has been successfully applied to various tasks, including image

inpainting, style transfer, and semantic segmentation. Applying the model's abilities in the context of architectural design, we use the instruct-Pix2Pix variant of the model to generate floorplan layouts from input images depicting footprints (the overall area of the apartment) and wall configurations. By training the unique conditional GAN (C-GAN) architecture, instruct-Pix2Pix learns to map input images to corresponding coloured-representative layouts of the ground truths, effectively capturing the underlying spatial relationships and architectural features.

In our study, we present a review of *ArchiGAN* and *Pix2Pix* in the context of generating apartment building layouts. We investigate their respective architectures, approaches, training methodologies, and performance characteristics. Additionally, we discuss and evaluate our approach to implementing the instruct-Pix2Pix model and assess its efficacy in generating layouts adhering to design constraints and aesthetic principles. Through our research experiment and assessment, we aim to review the strengths, limitations, and potential applications of deep learning and GAN-based approaches in architectural design.

# 2 Literature Review

## 2.1 Architectural Drawings Recognition and Generation Through Machine Learning

The combination of machine learning and architectural design has attracted increasing attention in recent years, driven by the promise of automating labour-intensive tasks and augmenting designers' capabilities. This literature review examines key studies and developments in the field, with a focus on architectural drawing recognition and generation through machine learning.

Early research in architectural drawing recognition primarily relied on rule-based systems and image-processing techniques. The review noted a key method for segmenting architectural floor plans using geometric primitives and line detection algorithms. While effective in specific contexts, these approaches often struggled with complex drawings and lacked adaptability to diverse architectural styles.

Now, the emergence of deep learning has revolutionized architectural drawing recognition by enabling end-to-end learning from raw image data. Deep learning frameworks for floor plan recognition demonstrated superior accuracy as compared to traditional methods. By using CNN's to extract features and classify floor plan elements such as rooms and walls. This approach marked a significant advancement in the field.

Building upon this research, recent studies have explored the application of conditional generative models, particularly GANs, for layout generation. The authors explored proposed solutions that outlined an interactive design system called Sketch2Layout, which employed a conditional GAN to generate floor plans from rough sketches drawn by users. The system included rapid prototyping and design exploration, showcasing the true power of GANs in supporting this creative process of architectural design.

The different papers referenced in this study presented an all-inclusive methodology for both architectural drawing recognition and generation. By leveraging CNNs for

recognition and conditional GANs for generation, the authors offer a creative approach to automated architectural design tasks. Their work builds upon past research while addressing the main limitations, such as the need for manual annotation and the lack of contextual awareness in generative models.

Furthermore, the article puts light to the diverse nature of research in this domain, drawing insights from computer vision, machine learning, deep learning and architectural practice. It shows the collective efforts between architects and data scientists in harnessing technology to re-imagine the design process.

The methodology outlined in the paper "Architectural Drawings Recognition and Generation through Machine Learning" discusses a well-studied approach to architectural drawing recognition and generation. They used a method which integrated CNNs for recognition and C-GANs for generation, offering a one-of-a-kind solution to automate architectural design tasks.

### Methodology:

- *Architectural Drawing Recognition*: The recognition process begins with preprocessing the architectural drawings to extract relevant features and normalize the quality of raw input data. Convolutional Neural Networks (CNNs) are employed as the primary recognition model due to their effectiveness in image classification tasks. The CNNs are trained on annotated datasets comprising various architectural elements such as walls, doors, windows, and rooms. Transfer learning techniques may be utilized to leverage pre-trained CNN models, thereby reducing the need for extensive manual annotation and accelerating the training process. The trained CNN model is then applied to recognize and classify architectural elements within input drawings, providing a structured representation of the design.
- *Architectural Drawing Generation*: The generation phase involves the use of conditional Generative Adversarial Networks (GANs), a deep learning architecture capable of generating realistic images conditioned on input data. The GAN framework consists of a generator network, which synthesizes architectural drawings based on learned representations, and a discriminator network, which distinguishes between real and generated drawings. To train the conditional GAN, a dataset of paired examples is required, comprising ground truth architectural drawings and corresponding contextual information. During training, the generator network learns to generate plausible architectural drawings that closely resemble the ground truth examples, while the discriminator network learns to differentiate between real and generated drawings. The training process involves iteratively optimizing the parameters of both networks through adversarial training, where the generator aims to deceive the discriminator, and the discriminator strives to accurately classify the generated drawings.

### Challenges Faced:

- Annotating large-scale datasets for training CNNs can be labor-intensive and time-consuming, posing a challenge to the scalability of the recognition model.

- Ensuring the accuracy and fidelity of generated architectural drawings remains a significant challenge, particularly in capturing fine-grained details and maintaining architectural coherence.
- Balancing the trade-off between exploration and exploitation in the generation process is essential to prevent the model from generating overly repetitive or unrealistic designs.
- Integrating the generated drawings into the architectural design workflow and addressing any discrepancies between human and machine-generated designs present additional challenges in real-world applications.

In summary, the publication reviewed highlights the transformative potential of machine learning in architectural design, offering novel approaches to drawing recognition and generation that empower designers and enrich the built environment. Future research directions may include refining the accuracy and robustness of recognition models through larger and more diverse datasets. Additionally, advancing generative models to incorporate user preferences, design constraints, and historical precedents could enhance their appeal to sceptics in the industry. Despite various challenges, the researchers' interdisciplinary approach and innovative methodologies pave the way for future advancements in this rapidly evolving field.

## 2.2 AI + Architecture — Towards a New Approach

The paper "AI + Architecture — Towards a New Approach" provides a comprehensive exploration of the intersection between artificial intelligence (AI) and architecture, offering insights into how AI technologies can revolutionize architectural design processes. The literature review presented in the paper synthesizes existing research and discusses key developments, challenges, and future directions in the field, shedding light on the transformative potential of AI in architecture.

### Key Themes and Developments:

The literature review identifies several key themes and developments that have shaped the intersection of AI and architecture. One prominent theme is the emergence of generative design techniques, which leverage AI algorithms to explore vast design spaces and generate novel architectural solutions. The journal discusses the role of genetic algorithms, neural networks, and other AI-driven approaches in enabling designers to explore alternative design possibilities and optimize architectural performance metrics. Another important theme is the integration of AI technologies into architectural practice, including parametric modelling, computational optimization, and robotic fabrication. These themes are backed by named case studies and real-world applications where AI tools and algorithms have been deployed to streamline design workflows, enhance collaboration, and facilitate the realization of complex architectural forms.

### Discussing the Approach:

- *Convolutional Neural Networks (CNNs) for Recognition:* The authors leverage CNNs, a type of deep learning model well-suited for image recognition tasks, to recognize architectural elements such as footprints and walls in input images. CNNs are trained on a labelled dataset of architectural drawings to learn the features and patterns associated with different elements.
- *Generative Adversarial Networks (GANs) for Generation:* To generate architectural layouts similar to ground truths, the authors employ GANs, a framework consisting of two neural networks – a generator and a discriminator – trained in an adversarial manner. The generator learns to produce synthetic images, while the discriminator learns to distinguish between real and fake images. Through iterative training, the generator improves its ability to generate realistic architectural layouts.

### Methodology:

- *Data Collection and Preprocessing:* The author gathered a dataset of architectural drawings, including ground truth layouts and corresponding footprints and walls. They preprocessed the data to ensure uniformity in size, resolution, and format, preparing it for training the machine learning models.
- *Model Training:* The CNNs for recognition and the GANs for generation are trained on the preprocessed dataset. The CNNs learn to classify architectural elements in input images, while the GANs learn to generate synthetic layouts resembling the ground truths. Training involves optimizing model parameters through backpropagation and gradient descent to minimize classification and generation errors.
- *Evaluation:* The trained models are evaluated using various metrics to assess their performance in recognizing architectural elements and generating realistic layouts. Evaluation criteria may include accuracy, precision, recall, and qualitative assessments of generated images' fidelity to ground truths.

### Challenges Faced:

Despite the promise of AI in architecture, the author acknowledges several challenges and limitations. These include issues related to data availability, algorithmic bias, and ethical considerations surrounding the use of AI in design decision-making. Along with these issues, the key challenges to note were as follows:

- *Dataset Annotation:* Annotating architectural drawings with ground truth labels for training the recognition model can be labour-intensive and time-consuming. Challenges may arise in accurately labelling unconventional architectural designs, requiring the aid of domain-specific expertise and manual effort.
- *Model Complexity:* Developing effective CNN and GAN architectures capable of accurately recognizing and generating architectural layouts with respect to the regional style poses technical challenges. Designing neural network architectures with the appropriate depth, width, and complexity to capture the intricacies and important features of architectural drawings is crucial for ensuring accurate and comprehensible outputs.

- *Overfitting and Generalization:* Overfitting can negatively affect the model's ability to generate coherent plans that fit the footprint. The inability of the model to generalize on new inputs can significantly impact its use case. To combat this flaw, techniques such as data augmentation, regularization, and cross-validation can be employed to mitigate overfitting and improve model generalization.
- *Computational Resources:* Extensively training CNNs and GANs on large datasets of high-resolution architectural drawings may require significant computational resources, including powerful GPUs and ample memory. Time and access restrictions for suitable computing infrastructure and efficient training algorithms are essential limitations in conducting experiments effectively.

The article concludes with a discussion of future directions and opportunities for research and innovation in AI and architecture. The author highlights the potential for AI to enable more sustainable, resilient, and human-centric architectural designs, citing advancements in computational design optimization, generative adversarial networks (GANs), and reinforcement learning. Additionally, the article strongly supported opportunities for continued experimentation, exploration, and knowledge-sharing within the architectural community to harness the full potential of AI technologies in shaping the built environment.

# 3 Methodology

Our methodology for training the pix2pix GAN model involved several key steps to ensure effective learning and generation of realistic architectural floor plans. Since the quality and diversity of the training data are paramount for the success of any deep learning model, we acquired a comprehensive dataset from HuggingFace. This dataset is comprised of eight parquets, each containing about 4000 items, which were subsequently divided into five distinct directories. These directories included images and texts representing various aspects of architectural floor plans, such as the basic footprints, colour-segmented floor plans, contrast-stretched representations of the walls, descriptions of the floorplans, and ground-truth floorplans. This variety ensures that the model learns a wide range of features for recognizing floor plan designs. Before feeding the data into the model, we pre-process it to ensure consistency and optimize training. This step involves resizing images to a uniform format, normalizing pixel values, and converting data types for uniformity. Though our approach did not require us to enhance the dataset's diversity, we can further improve the model's generalizability by implementing data augmentation techniques like random cropping, flipping, or rotations to artificially expand the dataset by creating variations of the existing data, preventing overfitting and improving the model's performance on unseen data. To aid in preprocessing the dataset, we leveraged the TensorFlow and Keras libraries to establish a robust pipeline for training the generator and discriminator components of the instruct-Pix2Pix GAN model.

For the architectural floorplan generation task, we adopted a U-Net-based architecture for our generator model due to its effectiveness in image-to-image translation tasks. The U-Net architecture, resembling the letter "U", consists of a contracting path

(encoder) and an expansive path (decoder) connected by skip connections. The sequential encoder layers downsampled the input image, extracting features at various scales, while the alternating decoder layers comprised of concatenating and upsampling the extracted features, reconstructing the image and incorporating contextual information from the encoder through skip connections and the final *Conv2DTranspose* layer. This structure allows the generator to capture both global and local features, crucial for generating detailed and realistic floor plans. Conversely, we employ a *PatchGAN* discriminator, a convolutional neural network that focuses on classifying individual image patches as real or fake. By scrutinizing local patches, the *PatchGAN* discriminator encourages the generator to generate realistic details at a finer scale, resulting in cohesive and visually appealing floor plans. Each discriminator block consists of a convolution layer followed by batch normalization and a Leaky ReLU activation function to mitigate the risk of layer inactivity.

### Loss Functions:

- *Generator Loss (LG):* The generator loss is typically a combination of L1 loss and adversarial loss.
- *L1 Loss:* Measures the pixel-wise difference between the generated image (G(x)) and the ground truth image (y):

$$L_{L1} = \frac{1}{N}\sum_{i=1}^{N}|y_i - G(x_i)| \tag{1}$$

- *Adversarial Loss:* Measures how well the discriminator (D) can distinguish between real and generated images:

$$L_{adv} = \frac{1}{N}\sum_{i=1}^{N} log(1 - D(G(x_i))) \tag{2}$$

- *Total generator Loss:*

$$LG = \lambda_{L1}L_{L1} + \lambda_{adv}L_{adv} \tag{3}$$

where $\lambda_{L1}$ and $\lambda_{adv}$ are weights to balance the contributions of each loss component.

- *Discriminator Loss(LD):* The discriminator loss is typically a sigmoid cross-entropy loss:

$$LD = \frac{1}{N}\sum_{i=1}^{N}[y_i log(D(y_i)) + (1 - y_i) log(1 - D(G(x_i)))] \tag{4}$$

where $y_i$ is 1 for real images and 0 for generated images.

Before starting the training process, we defined appropriate loss functions for both the generator and discriminator components. We utilize the combination of L1 loss and adversarial loss for the generator. The L1 loss encourages the generated image to be close to the ground truth in the pixel space, ensuring the overall layout and structure are preserved. The adversarial loss, on the other hand, was formulated as a sigmoid cross-entropy loss between the generated images and an array of ones, signifying genuine images. This pushes the generator to create images that are indistinguishable

from real floor plans, enhancing the realism and visual quality of the output. The discriminator loss evaluated the disparity between and generated images, with the *real_loss* function computing the sigmoid cross-entropy loss against an array of ones (real images), and the *generated_loss* computing the loss against an array of zeroes (fake images). The total loss is then calculated by summing the values of the *real_loss* and *generated_loss*.

For optimizing the generator and discriminator models during training, we utilized the Adam optimizer function from the TensorFlow-Keras library. The Adam optimizer is known for its efficiency in training NNs due to its adaptive learning rate and momentum estimation properties ensure efficient convergence and stable training. Additionally, we can explore hyperparameter tuning strategies such as learning rate scheduling or grid search to further optimize the training process and improve the model's performance in future tests. For our current experiment, we set a lower learning rate to ensure more stable convergence and smaller step sizes during optimization despite longer training times. We also set the exponential decay rate (*beta_1*) to 0.5 to balance the influence of the current and past gradients during optimization.

### Adam Optimizer:

The Adam optimizer updates the model parameters using the following equations:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{5}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{6}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{7}$$

$$\hat{v}_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \tag{8}$$

where:

- $m_t$ and $v_t$ are moving averages of the gradient and squared gradient, respectively.
- $\beta_1$ and $\beta_2$ are exponential decay rates for the moment estimates.
- $\eta$ is the learning rate.
- $\epsilon$ is a small constant to prevent division by zero.
- $g_t$ is the gradient at time step t.
- $\theta_t$ are the model parameters at time step t.

In configuring the training procedure, our objective was to train the model to generate floorplans based on input footprints. We executed 10,000 training steps with a batch size of 1, ensuring meticulous attention to detail. At every 5000 steps, we saved outputs and checkpoints to monitor the model's progress. During training, each input example was used to generate an output, with both the input image and the generated image fed into the discriminator for evaluation. The discriminator received two sets of inputs: the first set consisting of the input image and the generated image, and the second set containing the input image and the target image. Following this, we computed the generator and discriminator losses, followed by the calculation of

the gradients with respect to both generator and discriminator variables, which were then applied to the optimizer for parameter updates.

This comprehensive methodology facilitated the effective training of the instruct-Pix2Pix GAN model, allowing it to learn and generate realistic architectural floorplans based on input footprints.

# 4 Potential Improvements

While the methodology discussed provides a strong foundation for training the pix2pix GAN model to generate custom floorplans, several potential shortcomings and areas for improvement should be taken into consideration. Exploring alternative architectures or modifying the existing U-Net-based generator could be a possible option for improving the model's ability to capture relevant features and spatial relationships within the floorplan layouts. Additionally, using data augmentation techniques such as rotation, scaling, or flipping could enrich the existing training dataset and improve the model's generalization capabilities for varying layouts. Another method for improvement involves fine-tuning hyperparameters, to experiment with varying learning and decay rates, batch sizes, and training steps to optimize the training dynamics and convergence speeds. Furthermore, leveraging advanced regularization techniques like dropout or spectral normalization could mitigate overfitting and enhance the model's robustness. Lastly, incorporating adversarial training strategies, such as progressive growing or Wasserstein GANs, could refine the generator's output quality and yield floorplans closer to the ground truth. By iteratively refining the training process and incorporating these enhancements, we can foster significant improvements in the model's performance and fidelity of generated floorplans.

# 5 Conclusion

In conclusion, this research provides a way to use the power of deep learning and the abilities of the 'instruct-Pix2Pix GAN' model, to explore architectural design by automating the generation of a suitable floorplan given a footprint. Through extensive exploration of existing literature, we have gained valuable insights into the upcoming applications of AI in architecture, highlighting its great hidden potential and the emergence of innovative approaches like *ArchiGAN* and *AI + Architecture*. These works highlight the critical role of implementing machine learning methods to streamline design workflows, generate creative layouts, and unlock novel design possibilities.

Our project delved into the deep workings of the instruction-based Pix2Pix model and its GAN architecture. The motivation behind researching this particular style of GAN architecture came from the interesting approach to generating images in various formats whilst still being proficient enough to preserve their key features. By gathering insights from prior studies, projects and applications, using the instruct-Pix2Pix model to generate custom floorplans given their footprints as the condition was a new challenge to attempt. This approach allowed us to verify and rationalize the model's efficacy in this application of machine learning in architecture.

Our methodology, despite its simplicity, outlines our limited approach to training the instruct-Pix2Pix GAN model using a dataset comprising floorplan footprints,

coloured representations of the areas, outlines of the floorplan walls, descriptions of the floorplans and the corresponding ground truth images. Leveraging the TensorFlow 2 and Keras 3 libraries, we constructed a robust pipeline for our training method, with a U-Net-based generator and PatchGAN-based discriminator, to iteratively refine the model's output. To ensure the effectiveness of the training, we incorporated appropriate loss functions, optimizers, and training protocols to optimize the model's performance and generate floorplans that closely resemble the ground truth.

While our methodology lays a solid approach for training the instruct-Pix2Pix model, there are several potential improvements and opportunities for future research. A few of these advancements include exploring alternative architectural designs, augmenting the training dataset with diverse and regional samples, and fine-tuning hyperparameters. These techniques, once implemented, could vastly enhance the model's efficacy and generation capabilities. Moreover, integrating advanced regularization techniques, adversarial training strategies, and increasing the number of hidden layers will further refine the model's output quality.

In essence, our project allowed us to delve deep into the techniques pivotal towards harnessing the potential of machine learning in architectural design applications. By leveraging the powerful GAN capabilities of the instruct-Pix2Pix model and iteratively refining our methodologies, we studied the unique approach to generating floorplans efficiently and creatively. Through continuous research and innovation in this novel application, we note the formative future of AI in architecture and its ability to shape the built environment in new and transformative ways.

# 6 Figures and Diagrams

The list of the Figures are as follows:

1. Fig1 : Representation of the Generator Architecture [6]
2. Fig2 : Representation of the Discriminator Architecture [7]
3. Fig3 : Visualization of the Training Procedure of the Generator[8]
4. Fig4 : Visualization of the Training Procedure of the Discriminator[9]

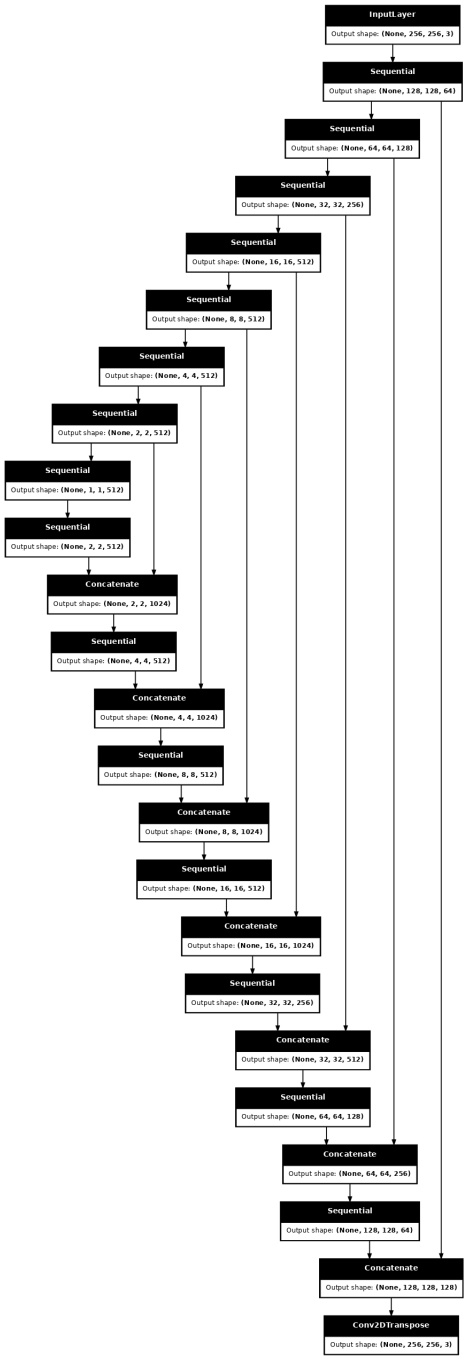


**Fig. 1** Visualization of the Generator Architecture [6]

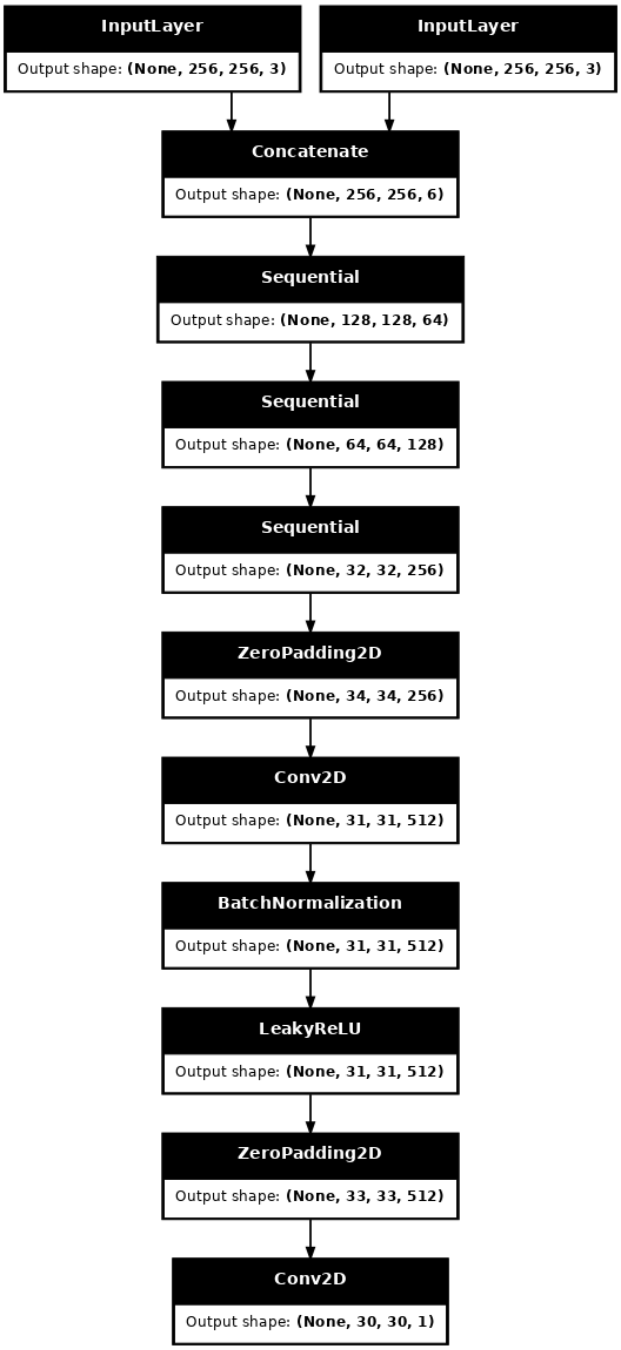


**Fig. 2** Visualization of the Discriminator Architecture [7]

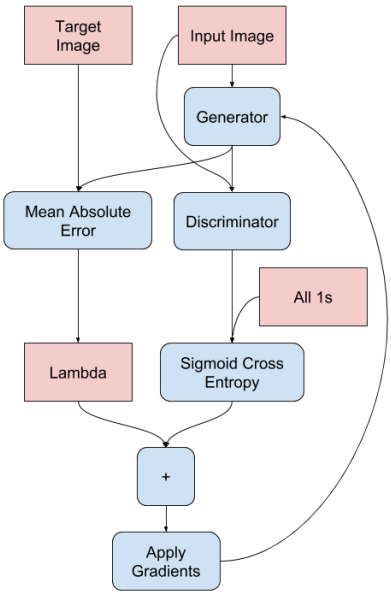


**Fig. 3** Visualization of the Training Procedure of the Generator [8]

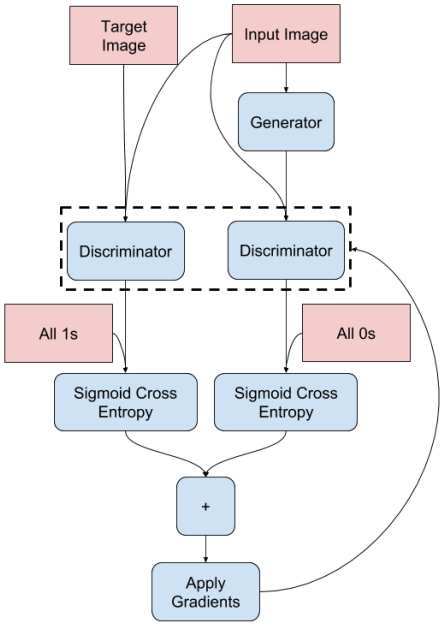


**Fig. 4** Visualization of the Training Procedure of the Discriminator [9]

# References


[1] "Pix2pix: Image-to-image translation with a conditional Gan : Tensorflow Core," TensorFlow, https://www.tensorflow.org/tutorials/generative/pix2pix (accessed Apr. 30, 2024).

[2] S. Leng et al., "Tell2Design: A dataset for language-guided floor plan generation," Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2023. doi:10.18653/v1/2023.acl-long.820

[3] S. Chaillou, "Ai + architecture: Towards a new approach," thesis, 2019

[4] S. Chaillou, "ArchiGAN: a Generative Stack for Apartment Building Design," NVIDIA Developer Technical Blog,

[5] zimhe, "huggingface/datasets/zimhe/sudo-floor-plan-12k." https://huggingface.co/datasets/zimhe/sudo-floor-plan-12k, Sep. 2023

[6] "Generator Architecture : Pix2pix: Image-to-image translation with a conditional Gan: Tensorflow Core", TensorFlow, https://www.tensorflow.org/static/tutorials/generative/pix2pix_files/output_dIbRPFzjmV85_0.png

[7] "Discriminator Architecture : Pix2pix: Image-to-image translation with a conditional Gan: Tensorflow Core", TensorFlow, https://www.tensorflow.org/static/tutorials/generative/pix2pix_files/output_YHoUui4om-Ev_0.png

[8] "Generator Training Procedure : Pix2pix: Image-to-image translation with a conditional Gan: Tensorflow Core", TensorFlow, https://github.com/tensorflow/docs/blob/master/site/en/tutorials/generative/images/gen.png?raw=1

[9] "Discriminator Training Procedure : Pix2pix: Image-to-image translation with a conditional Gan: Tensorflow Core", TensorFlow, https://github.com/tensorflow/docs/blob/master/site/en/tutorials/generative/images/dis.png?raw=1

[10] I. Goodfellow et al., "Generative Adversarial Networks," Communications of the ACM, vol. 63, no. 11, pp. 139–144, Oct. 2020. doi:10.1145/3422622

[11] N. Milosevic, "Convolutions and Convolutional Neural Networks," Introduction to Convolutional Neural Networks, 2020. doi:10.1007/978-1-4842-5648-0_12

[12] O. Ronneberger, "Invited talk: U-net convolutional networks for biomedical image segmentation," Informatik aktuell, pp. 3–3, 2017. doi:10.1007/978-3-662-54345-0_3