Exploring Alternative Architectures For The Pix2Pix Model

Maitar Asher, Ahuva Bechhofer, Shmuel Berman

I. Abstract

Building upon Isola et al.'s pioneering work in "Image-to-Image Translation with Conditional Adversarial Networks" [1], this project focuses on the Pix2Pix software associated with their paper. By exploring modifications and refinements to their CGAN network architecture, including variations in the generator's and discriminator's design, our objective is to delve deeper into the underlying mechanisms contributing to the remarkable efficacy of Pix2Pix. Through systematic experimentation, we aim to reveal the nuanced impacts of these adjustments on the model's performance and output quality. Our findings reveal that changing the generator architecture with respect to the "U-Net" skip connections did not enhance peak model performance although it did improve training speed. The incorporation of multi-scale discriminators in diverse configurations demonstrated promising outcomes in both output quality and quantitative measurements.

II. Introduction

Image-to-image translation is a major task in computer vision and graphics, spanning a wide range of applications such as converting sketches into lifelike photographs and infusing color into black and white images. The year 2016 marked a pivotal milestone in the domain of image-to-image translation with the publication of "Image-to-Image Translation with Conditional Adversarial Networks" by Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros [1]. Their work demonstrated a substantial advancement in this field by effectively utilizing Generative Adversarial Networks (GANs) [2] in a conditional setting. Conditional Generative Adversarial Network (CGAN) [3] emerged as a robust framework for tackling this challenge, offering a versatile approach that not only learns the mapping from input to output images but also the loss function to facilitate this mapping. This present work extends the foundations laid by Isola et al., delving further into the exploration of image-to-image translation with Conditional Generative Adversarial Networks (CGAN).

III. Related Work

A. A review of GANs

Generative Adversarial Networks (GANs) [2] have carved a niche for themselves in the landscape of generative models, primarily due to their novel architecture and approach to training. At their core, GANs consist of two competing neural network models: a generator (G) and a discriminator (D). The generator attempts to create data that is indistinguishable from genuine data, while the discriminator evaluates the generated data against the actual data, acting as a critic to assess the authenticity. This adversarial process is akin to a forger trying to create a counterfeit painting and an art critic trying to detect the forgery. The original GAN framework operates by training the discriminator to maximize the probability of correctly labeling both real and fake data. Simultaneously, the generator is trained to minimize the log probability of the discriminator being correct. This simultaneous training results in a minimax two-player game, with the generator and discriminator improving iteratively through their competition.

B. A review of CGANs

The advent of Conditional Generative Adversarial Networks (CGANs) [3] marked a pivotal enhancement over their predecessor, the basic GAN framework. While GANs learn to generate data from a random noise vector, CGANs refine this process by conditioning the generation on auxiliary information. This conditioning could be anything from class labels to portions of data from other modalities, such as text descriptions or images in a different domain. This auxiliary information is fed into both the generator and the discriminator, modifying their respective input layers and thus, fundamentally altering the architecture and functioning of the networks. The generator in a CGAN is designed to create output that is not only realistic but also appropriate to the given condition. The discriminator, on the other hand, is tasked with evaluating the authenticity of the generated data, considering both the realism of the data and the fidelity of the data to the condition. This dual requirement significantly changes the dynamics of the adversarial training, as the discriminator now guides the generator more specifically towards a targeted output. The conditional aspect of CGANs also changes loss function. The discriminator's loss includes terms that account for the correctness of the condition, in addition to the authenticity of the data. The generator's loss, consequently, penalizes the failure to meet the condition. This structured loss function is key to CGANs' ability to produce relevant and context-aware outputs. CGANs, with their structured output space, have been particularly successful in tasks that require a high degree of specificity, such as photo-realistic image synthesis conditioned on semantic labels, and style transfer where the output is conditioned on features of a particular style image. These networks have not only shown a remarkable ability to adhere to the conditions but have also been found to be more stable during training compared to traditional GANs, mainly due to the additional information that helps to anchor the generative process.

*We utilized the codebase from https://github.com/akanametov/pix2pix as a foundation and incorporated our custom modifications. Our code can be found at https://github.com/ahuvabec/NNDL-Research-Project

C. A review of Pix2Pix

Pixel-to-pixel translation is the idea that given an image, each individual pixel gets converted into another pixel. This model utilizes a CGAN architecture to synthesize an image from one space into another under a learned conditional setting. Refer to Figure 1, adapted from the Pix2Pix paper [1], for an example of training where the condition is on the input edge map. This implementation utilizes a PatchGAN Discriminator which takes an image and for every N*N patch, it predicts if each pixel in it has been generated or is it real. This added constraint helps the model be more attentive to sharp high-frequency details.

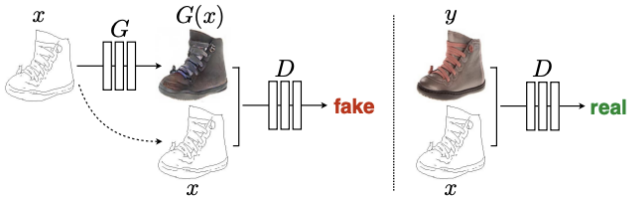


Fig. 1: Conditional GAN Training for Edge-to-Photo Mapping

IV. Materials and Methods

A. Dataset

In this study, we deliberately concentrated our efforts on a focused training set to navigate the challenges posed by constraints in time and resources. Our training dataset comprises a modest 400 images, drawn from the building facade (face of a building) dataset which aligns with the dataset utilized in the original paper. The decision to employ this specific dataset stems from its alignment with the original research and the acknowledgment, as stated in the paper, that "decent results can often be obtained even on small datasets" [1]. This strategic selection allows us to explore the model's capabilities within resource limitations while leveraging the insights provided by the original work. The specific mapping used in our analysis is visually represented in Figure 2.

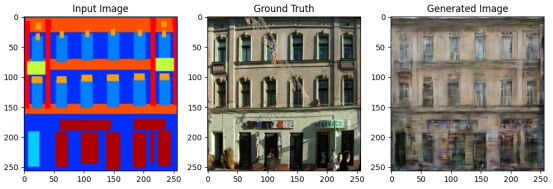


Fig. 2: Example of Input, Ground Truth, and Generated Image by Base Model

B. Methodology

The central objective of this project is to conduct experiments and evaluate various architectural configurations to gain deeper insights into the underlying mechanisms of the CGAN model. In particular, we explored:

1) Skip Connection variation: The generator's architecture, as described in the paper we are exploring, adopts a 'U-Net' structure. There are 8 encoder layers which are mirrored by 8 decoder layers, each respectively performing a convolution. Each encoder layer contains a "skip connection,", where its output is copied and concatenated, to the mirror layer on the decoder side of the network. The structure of the network, which has the most compact representation in the middle and thus has a bottleneck at that point, contains these skip connections so that high-granularity data is not lost through the network. This architecture facilitates early layers learning the most essential information and allows later layers to use information contained in the original image even if it is not present in the compact representation.

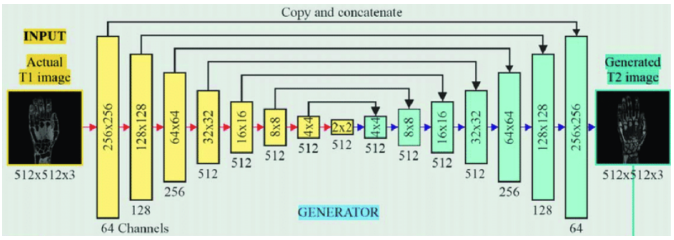


Fig. 3: U-Net architecture in pix2pix. The black arrows indicate skip connections, which the base model implements as copy and concatenate operations.

We aim to investigate alternatives to skip connections, such as addition and multiplication in pursuit of a more sophisticated architecture capable of capturing more complex mappings from input to output. We aim to compare not only the ultimate performance of the model but also the training time and training loss over time to compare how different architectures affect the training process. We compare average epoch time, training loss over time, the variability of the validation loss, and the FID score (Frechet Inception Distance, explained in the 'Results' section).

2) Multi-scale Discriminator: The paper we are exploring investigates the use of a GAN discriminator architecture that classifies NxN patches within an image as real or fake. This discriminator functions as a feature map and is applied convolutionally across the image. In an effort to enhance the model's ability to capture both local and global structures, we propose an alternative approach: employing multiple discriminators that assess various scales of the image. In our exploration of this alternative strategy, we conducted a series of experiments involving what we refer to as a "Multi-scale Discriminator" framework. This framework incorporates two discriminators, labeled "small" and "large," each utilizing distinct patch sizes. The rationale behind this approach is to provide the network with a means to capture both local and global information effectively. Specifically, the discriminator with a smaller patch size focuses on acquiring more detailed local information, while the one with a larger patch size is geared towards capturing broader, global structures. To implement this concept, we introduced an additional hyperparameter denoted as **ld_alpha** (large discriminator alpha), which governs the influence of the larger discriminator in the overall training

process of the generator (the discriminators themselves are trained independently). For a visual representation of the Multi-scale training of the generator and discriminators refer to Figures 4 and 5.

Exploration of Techniques:

In our pursuit of optimizing the multi-discriminator architecture, we experimented with two distinct techniques to balance the contributions of the small and large discriminators throughout the training process:

- Consistent Weights Approach:
  Throughout the training, we applied a consistent weight for the large discriminator denoted by ld_alpha and a complementary weight for the small discriminator (1 - ld_alpha). This strategy maintains a fixed balance between the influence of the small and large discriminators, allowing the model to continuously incorporate both local and global information. The objective function of the generator can be expressed as

$$G^* = \arg\min_G \max_{D_\text{l}, D_\text{s}} \Big(\text{ld\_alpha} \cdot L_{cGAN}(G, D_\text{l}) + (1 - \text{ld\_alpha}) \cdot L_{cGAN}(G, D_\text{s}) + \lambda L_{L1}(G) \Big) \quad (1)$$

  which incorporates both discriminators with a static weighting factor.
- Dynamic Updates Approach:
  We initiated the training with a specific learning rate for the large discriminator and gradually decreased it over time. Simultaneously, the learning rate for the small discriminator increased progressively. This dynamic update mechanism aims to prioritize capturing more global information during the initial stages of training and gradually shift the focus toward the finer details provided by the small discriminator. The generator's objective function can be expressed as

$$G^* = \arg\min_G \max_{D_l, D_s} \Big((\text{ld\_alpha})^d \cdot L_{cGAN}(G, D_l) + (1 - (\text{ld\_alpha})^d) \cdot L_{cGAN}(G, D_s) + \lambda L_{L1}(G) \Big) \quad (2)$$

  where d is a parameter that increases by 1 every few epochs. The formula for the dynamic weights are:

$$\text{large\_discriminator\_weight} = \text{ld\_alpha}^\text{d}$$

$$\text{small\_discriminator\_weight} = 1 - \text{ld\_alpha}^\text{d}$$

  In Version 1, the d parameter starts with 0, and we implemented 5 updates to the weights. Consequently, if $ld\_alpha = 0.7$ the sets of weights for the large and small discriminators were $(1.0, 0.0), (0.7, 0.3), (0.49, 0.51), (0.343, 0.657), (0.2401, 0.7599)$. In Version 2, the d parameter starts with 1, and we implemented 2 updates to the weights. For the same $ld\_alpha = 0.7$ the set of weights were $(0.7, 0.3), (0.49, 0.51)$.

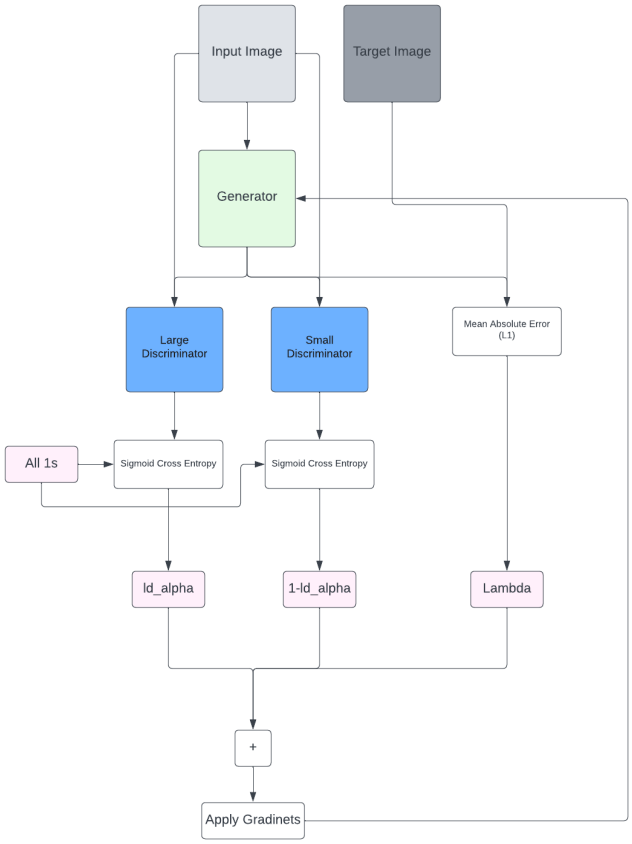


Fig. 4: Training Procedure for the Generator in the Multi-scale Discriminator Model (Consistent Weights Approach)

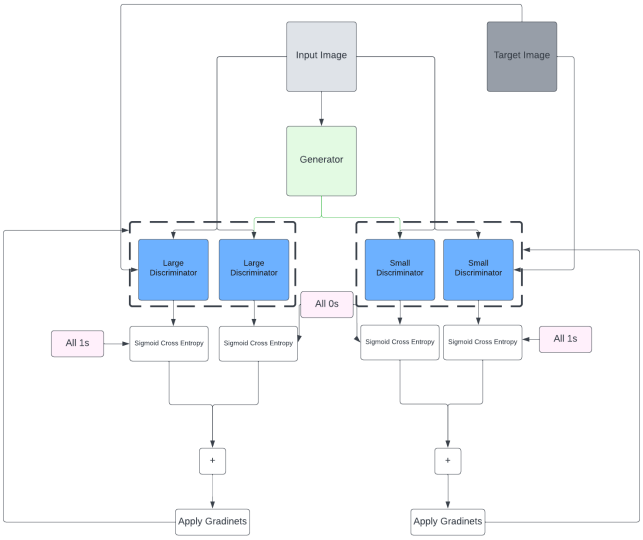


Fig. 5: Training Procedure for the Discriminators in the Multi-scale Discriminator Model

The subsequent section details the experimental setup, results, and implications of this novel multi-discriminator architecture. These experiments aim to shed light on the effectiveness of employing discriminators with varying scales in enhancing the spatial context awareness of the GAN model.

V. Results

We do not use validation or test loss as a metric to evaluate our models. Evaluating the performance of Generative Adversarial Networks (GANs), such as pix2pix, using traditional metrics like validation loss can be misleading and does not effectively represent the model's performance. Because the validation loss is a function of the disparity between the generator and discriminator, the generator's loss does not necessarily decrease as its performance improves. We confirmed this through analysis of the produced images.

For a comprehensive and quantitative assessment, we employed the **Fréchet Inception Distance (FID) score** (using Seitzer's codebase [4]). FID is a well-established metric in the field of generative models, particularly suited for tasks like image synthesis and generation. It quantifies the quality and diversity of generated images by measuring how closely they align with the distribution of real images. It's essential to note that FID scores typically range from 0 to positive infinity, with lower scores indicating better performance. A lower FID score implies that the generated images closely match the distribution of real images, signifying higher quality and diversity. While there isn't a universally defined threshold for what constitutes a "good" FID score, generally, scores closer to zero are desirable, suggesting a more faithful reproduction of real image characteristics by the generative model. In practice, a lower FID score reflects a higher degree of similarity between generated and real images, indicating superior performance in tasks such as image synthesis or generation.

A. Base model results

The following are the configurations we examined for the base model.

Configuration 1:

- Number of epochs: 200
- Batch size: 1
- Learning Rate: 0.0002
- Training Time: $\sim$ 17 seconds per epoch
- FID score: $186.1885, 188.0523$
  (under two independent experiments)

Configuration 2:

- Number of epochs: 32
- Batch size: 1
- Learning Rate: 0.0002
- Training Time: $\sim$ 5 seconds per epoch
- FID score: 292.1598

In accordance with the original paper's recommendations, Configuration 1 yields the most favorable results for the base model and serves as a benchmark for subsequent modifications. The choice of a batch size of 1 aligns with established practices of the Pix2Pix model and is consistent with common approaches in various image generation tasks. The lower FID scores achieved in configuration 1 further validate the appropriateness of this selection. The decision to conduct training for 200 epochs was influenced by our examination of existing sources, including the original paper, which did not advocate for early stopping—a departure from common practices. Figure 7 underscores the model's inability to determine when to stop based on validation loss, with fluctuations evident in the loss curves. Our investigation reveals that 200 epochs were deemed optimal for the facades dataset, especially considering the continued decline in generator loss as depicted in Figure 8. The reported training time in this paper is based on training conducted on a single Nvidia L4 GPU with 24GB GDDR6 memory.

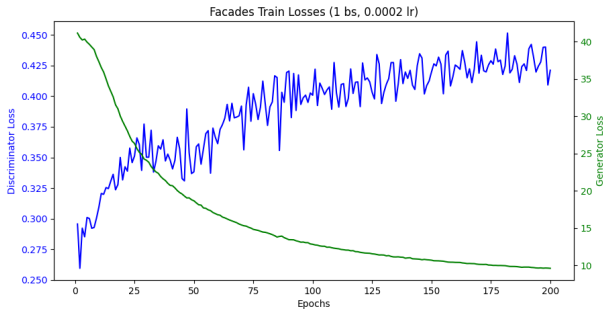


Fig. 6: Training Losses of Generator and Discriminator for Base Model Configuration 1

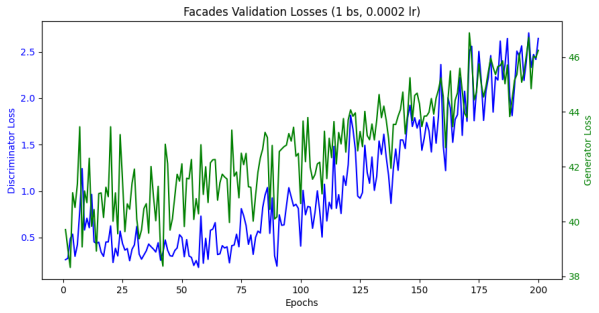


Fig. 7: Validation Losses of Generator and Discriminator for Base Model Configuration 1

B. Skip Connection variation results

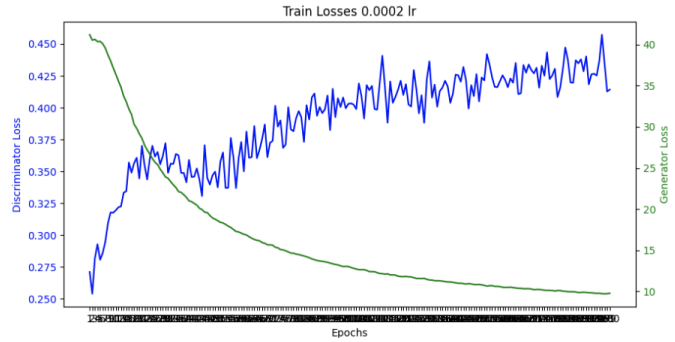
To find the effect of replacing the skip connections with add or multiply connections, we ran multiple experiments in which we changed the architecture to have add or multiply connections instead of concatenations. In an add connection, we do an element-wise addition of the encoder and decoder layer; in a multiply connection, we do an element-wise multiplication of the encoder and decoder layers.

In the first experiment, we simultaneously ran an experiment with skip connections, add-connections, and multiply-connections. As recommended by the original paper, we ran with a batch size of 1, a learning rate of 0.0002 and 200 epochs. The final results are summarized in Table 1, and Figure 8 has graphs of the generator and discriminator loss respectively accross the different architectures.

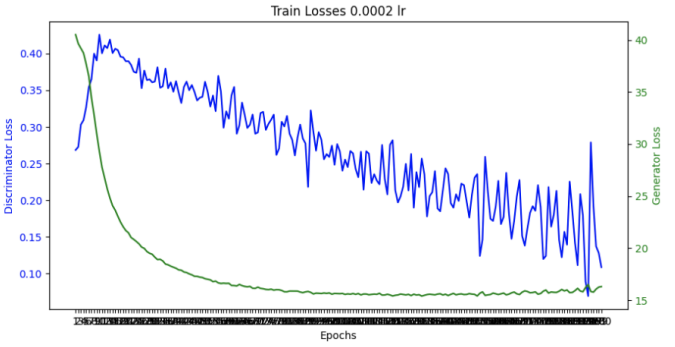
The average epoch time of the skip connection architecture was significantly longer– by around 15% – as compared to the

TABLE I: Experiment Parameters: Batch Size = 1, Learning Rate = 0.0002, Epochs = 200. All experiments were run simultaneously.

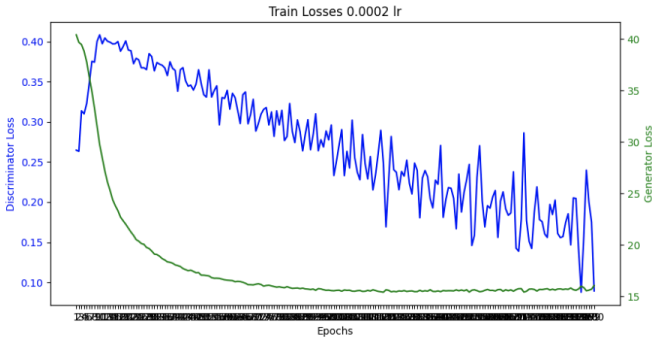
Connection Type	Average Epoch Time (s)	FID
Skip Connection	38.5159	188.0523
Multiply Connection	33.1366	384.0897
Add Connection	32.6704	367.7747



(a) Concatenation Connection



(b) Multiply Connection



(c) Add Connection

Fig. 8: Generator Skip-connection training loss over 200 epochs. Batch Size = 1, learning rate = 0.0002. Generator and discriminator loss are green and blue respectively.

multiply or add connections. This is expected, as the size of the network at the concatenation points is double as compared to an element-wise operation like addition or multiplication. This increased model complexity increases the time needed for back-propagation. Addition is a faster operation than multiplication, which explains the marginally faster average time in addition model as compared to the multiplication model.

The add-connection and multiply-connection models initially trained much faster than the skip-connection model, but the skip-connection model kept improving long after both the add and multiply models had ceased to improve either their FID score or training loss. For instance, after 20 epochs of training, the add and multiply models had a reached a training loss of 22 while the skip-connection model still was at 30. At 60 epochs, the models all have roughly equivalent training losses and FID scores– 17 and 400, respectively– but the similarities end there. Only the skip-connection model continues to improve.

During the initial 30 epochs of training, there was a consistent improvement in the performance of discriminators across all models. The long-term trend of discriminator loss diverges significantly between models employing concatenation connections and those with element-wise connections. In the latter models, the discriminator loss demonstrates a continual decline, indicative of an increasing ability of the discriminator to differentiate between real and synthetic data. Conversely, in models utilizing concatenation connections, the discriminator loss increases throughout the entire 200-epoch training period.

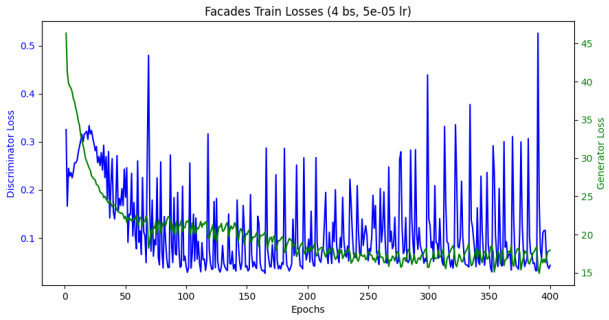
Given the unaltered architecture of the discriminator across these experiments, this phenomenon suggests a disparity in the element-wise skip-connection-generator's capacity to evolve and effectively challenge the discriminator. Particularly, the increase in discriminator loss in concatenation connection models may reflect a stagnation in generator performance, unable to adequately adapt or improve in response to the discriminator's enhanced discernment. This hypothesis is supported by recent GAN literature, which emphasizes the dynamic interplay between the discriminator and generator. Yang et al. [5] discuss the problems that result when a discriminator gets too good for the generator, and even argue that better performance can be attained if the discriminator knows when to scale back.

Our initial tests revealed the performance of add-connections and multiply-connections to be very similar. However, when we lowered the learning rate and increased the number of epochs, the performance of the add-connection architecture improved dramatically, as shown in Table II. The performance of the concatenation-connection architecture worsened. Examples of the actual images produces by experiments can be found in the 'Experiment Outputs' section in Figures 20 and 21.

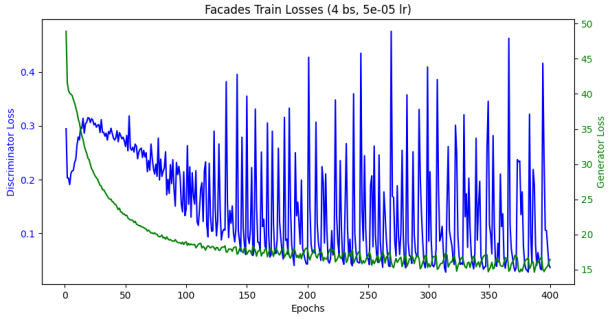
To confirm the worse performance on the concatenation-connection was due to the learning rate, we re-ran the experiment with a batch size of 1, which resulted in an FID of 210.700, suggesting the base model was underfit and could be trained further.

TABLE II: FID Scores at 250 and 400 Epochs (Batch Size - 4, lr = 0.00005)

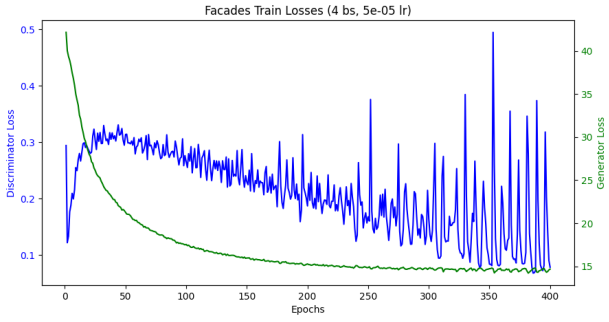
Skip Connection Type	FID at 250 Epochs	FID at 400 Epochs
Concatenation Connection	217.368	234.933
Add Connection	217.161	212.486
Multiply Connection	189.582	191.769



(a) Concatenation Connection



(b) Add Connection



(c) Multiply Connection

Fig. 9: Generator Skip-connection training loss over 400 epochs. Batch Size = 4, learning rate = 0.00005. Generator and discriminator loss are green and blue respectively.

1) Analysis: Our experiments suggest that all models can produce similar levels of output images, but that the concatenation-connection model is significantly easier to train and suffers less from both underfitting and overfitting, as the range of terminal FID was much smaller for this model between experiments. Additionally, our experiments suggest that the multiply-connection architecture learns the fastest, followed by the add-connection, followed by the concatenation connection.

That concatenation takes longer to train is intuitive because of increased model size and complexity. Multiplication skip connections can potentially provide a stronger signal for gradient propagation compared to element-wise addition. This is because multiplication can amplify the gradients if the values are greater than one, or attenuate them less than addition if the values are between zero and one. This amplification can lead to more significant updates during the backpropagation process.

Our research supports the usage of concatenation connections as skip connections, but also suggests that multiply or add connections can produce a good-enough model faster if training time is important.

C. Multi-scale Discriminator results

The following are the configurations we investigated for the Multi-scale Discriminator, encompassing a small discriminator patch size of 2 by 2 and a large discriminator patch size of 4 by 4. In all of these configurations, the number of epochs is set to 200, the batch size is 1, and the learning rate is 0.0002, mirroring the settings of configuration 1 of the base model. The average training time per epoch for the Multi-scale Discriminator under these settings was $\sim$ 35 seconds.

Configuration 1:

- Technique: dynamic updates approach (version 1)
- ld_alpha: 0.9
- FID score: 298.3568

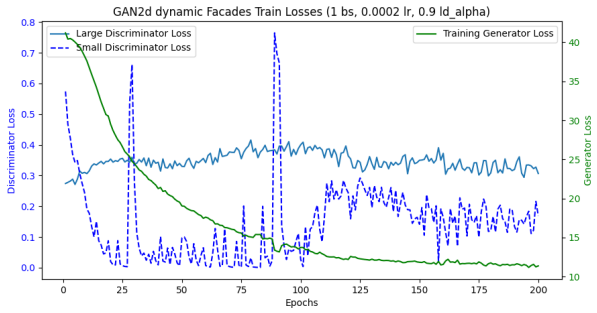


Fig. 10: Configuration 1 Training losses

Configuration 2:

- Technique: consistent weights approach
- ld_alpha: 0.7
- FID score: 184.5843

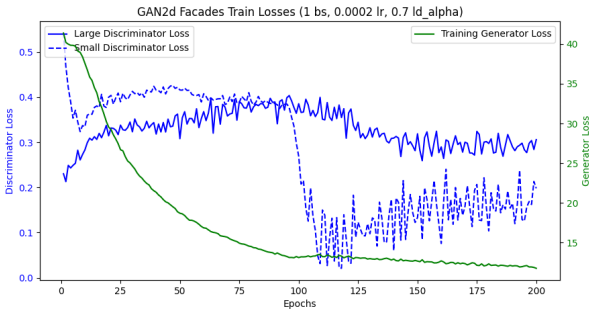


Fig. 11: Configuration 2 Training losses

***Configuration 3*:**

- Technique: dynamic updates approach (version 1)
- ld_alpha: 0.7
- FID score: 185.2853

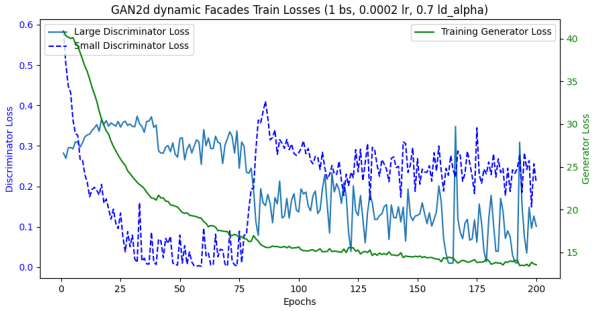


Fig. 12: Configuration 3 Training losses

Configuration 4:

- Technique: dynamic updates approach (version 2)
- ld_alpha: 0.7
- FID score: 194.8962

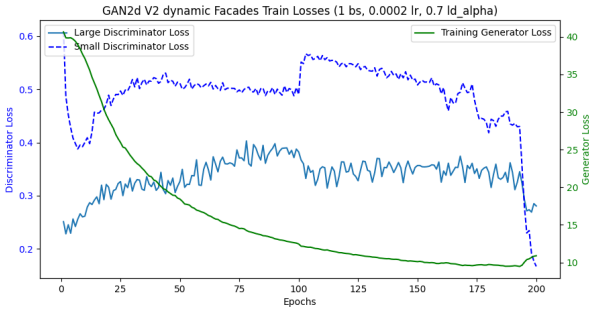


Fig. 13: Configuration 4 Training losses

Configuration 5:

- Technique: consistent weights approach
- ld_alpha: 0.5
- FID score: 190.4327

Configuration 6:

- Technique: dynamic updates approach (version 1)
- ld_alpha: 0.5
- FID score: 190.4333

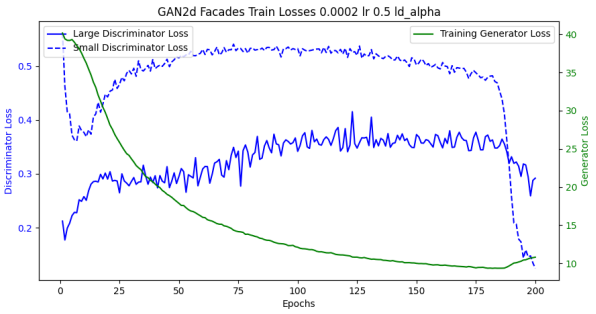


Fig. 14: Configuration 5 Training losses

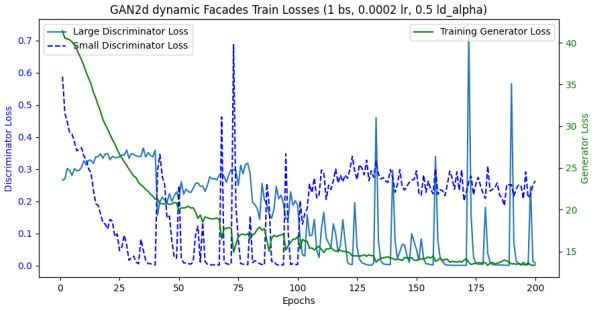


Fig. 15: Configuration 6 Training losses

Configuration 7:

- Technique: consistent weights approach
- ld_alpha: 0.3
- FID score: 186.9780

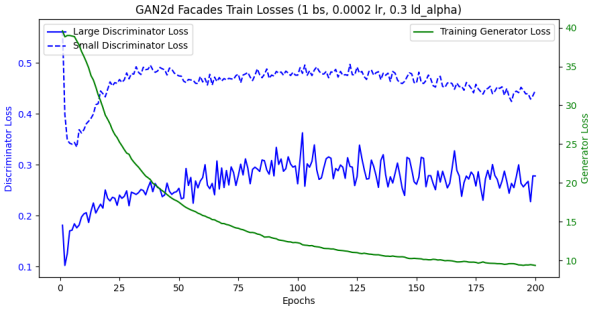


Fig. 16: Configuration 7 Training losses

Configuration 8:

- Technique: dynamic updates approach (version 1)
- ld_alpha: 0.3
- FID score: 295.5034

The outcomes of our experiments, detailed in the 'Experiment Outputs' Section, offer valuable insights into the performance of our models. Our analysis encompasses various metrics: the training loss, the FID score, and the qualitative aspects of the visual outputs. Notably, configurations 2 and 3 demonstrate lower FID scores compared to the base model, suggesting their generated outputs bear a closer resemblance to the input images. This resemblance indicates a more

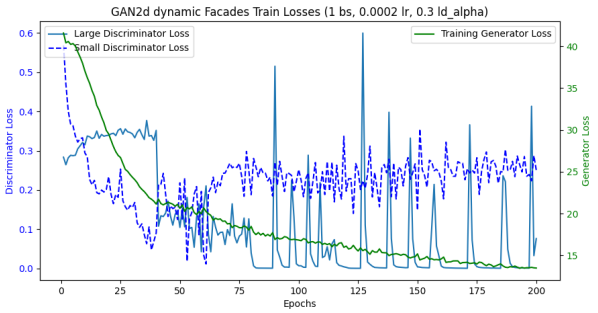


Fig. 17: Configuration 8 Training losses

accurate reproduction of the input, fulfilling our objective and underscoring these configurations' proficiency in capturing intricate details. Configurations 4, 5, and 6 yield the most aesthetically pleasing results, successfully overcoming the black outlines present in the ground truth images. This modification accounts for their relatively higher FID scores, as the peripheries in the ground truth are dark, whereas the generated images incorporate vibrant, contextually appropriate pixel hues. Moreover, the slight blurriness observed in these outputs which also contributes to the higher FID score, suggests these models have a better grasp of the global context, but lack in finer detail resolution.

Across all trials, the generator training loss consistently converged to a range between 10-15, implying that the models share a certain degree of similarity. This convergence hints at the potential for more radical research explorations.

Our other experiments, which explored different patch sizes, did not yield the anticipated improvements. It appears that further investigation into the interaction between patch sizes and configurations 4, 5, and 6 is warranted, as it may pave the way to a model that balances the capture of both global context and fine detail.

VI. Conclusion and Future Work

Our experiments with the proposed architectures are showing encouraging outcomes. Exploring various connection types, we unveil the inherent trade-offs—specifically, training velocity and hyper-parameter tolerance— and for our multi-discriminator setup, the strategy to grasp both the broader view and the finer details is working well, particularly in configurations 2 and 3 of the Multi-scale Discriminator experiments. There's more room for detailed exploration with these structures.

Instead of just using a single type of connection, future research could try and take advantage of multiplication connection's ability to amplify gradients and speed up the training process with concatenation connections that prevent vanishing gradients and ensure training is successful across a range of hyper-parameters. This could be accomplished by alternating concatenation connections and multiplication connections between mirrored layers.

As for the approach with multiple discriminators, further studies could investigate using more than two discriminators, experimenting with various patch sizes, and finding the best way to adjust the weighting dynamically. One idea we haven't tried yet is to change the value of ld_alpha during training, depending on the loss at each epoch. Digging deeper into these ideas could lead to even more impressive results.

References

[1] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arXiv preprint arXiv:1611.07004*, 2016.

[2] I. J. Goodfellow, M. Mirza, B. Xu, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *arXiv preprint arXiv:1406.2661*, 2014.

[3] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

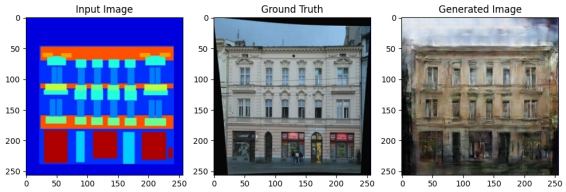
[4] M. Seitzer, "pytorch-fid: FID Score for PyTorch," https://github.com/mseitzer/pytorch-fid, August 2020, version 0.3.0.

[5] Author(s), "Improving gans with a dynamic discriminator," *arXiv*, vol. arXiv:2209.09897, 2022.

VII. Experiment Outputs

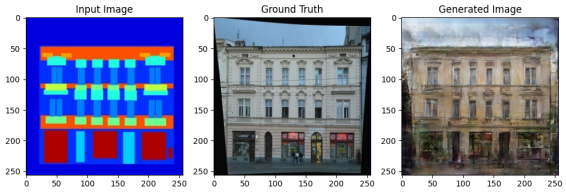
(a) Base model Output

(b) Configuration 1 Output



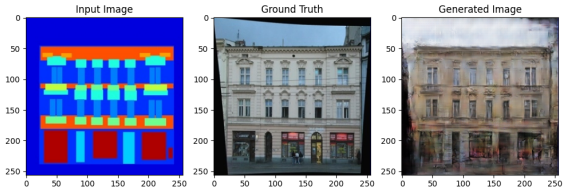
(c) Configuration 2 Output

(d) Configuration 3 Output



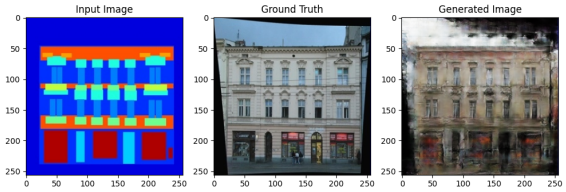
(e) Configuration 4 Output

(f) Configuration 5 Output



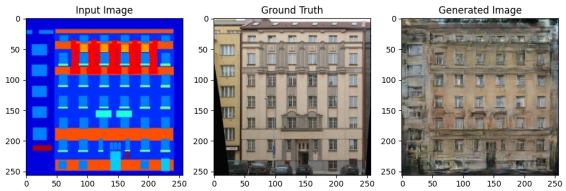
(g) Configuration 6 Output

(h) Configuration 7 Output

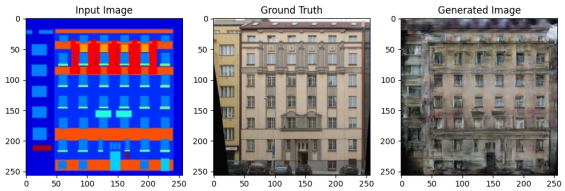


(i) Configuration 8 Output

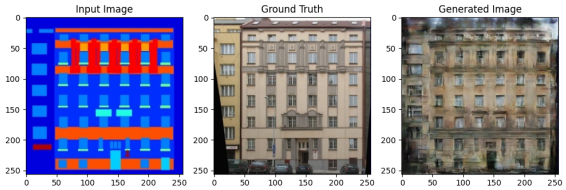
Fig. 18: Outputs from different configurations of the Multi-scale Discriminator



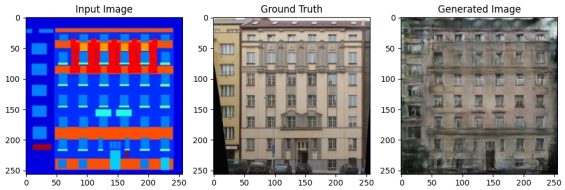
(a) Base model Output



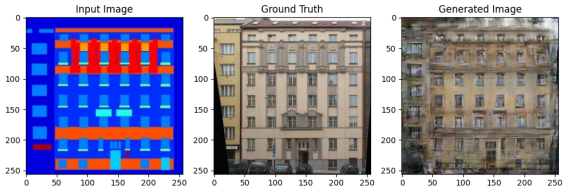
(b) Configuration 1 Output



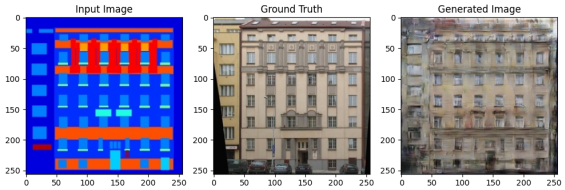
(c) Configuration 2 Output



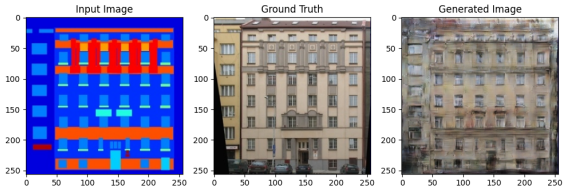
(d) Configuration 3 Output



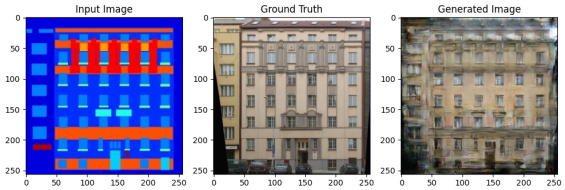
(e) Configuration 4 Output



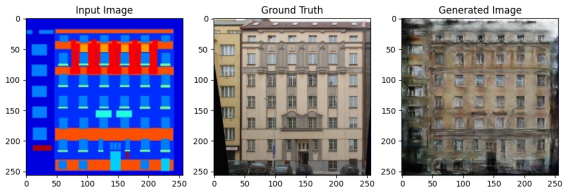
(f) Configuration 5 Output



(g) Configuration 6 Output

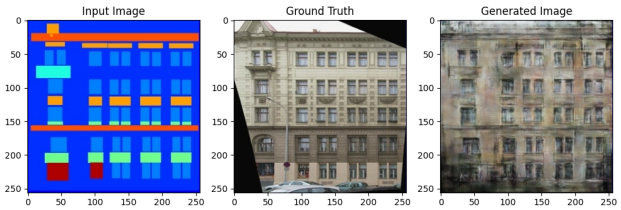


(h) Configuration 7 Output

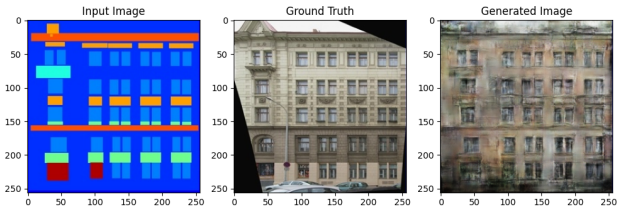


(i) Configuration 8 Output

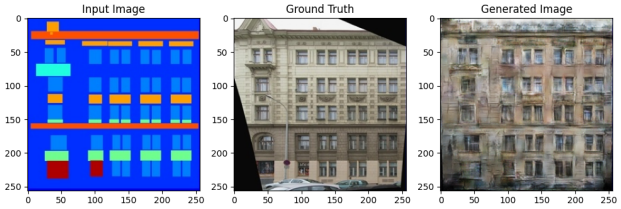
Fig. 19: Outputs from different configurations of the Multi-scale Discriminator



(a) Concatenation Connection

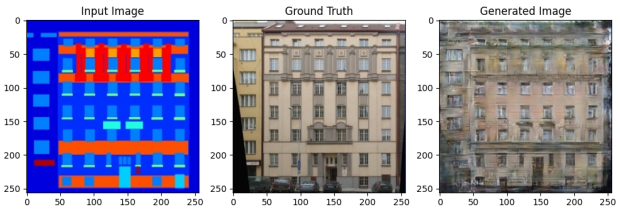


(b) Add Connection

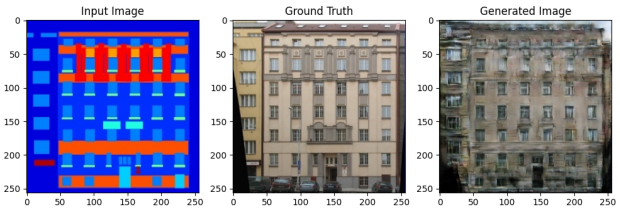


(c) Multiply Connection

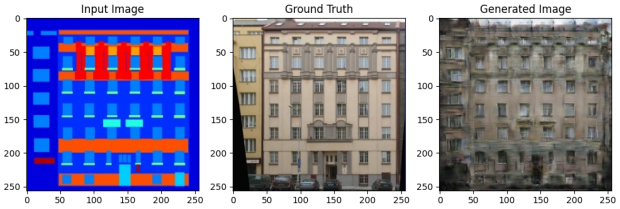
Fig. 20: Comparison of Concatenation, Add, and Multiply Connections at 400 Epochs (learning rate = 0.00005, Batch Size = 4)



(a) Concatenation Connection



(b) Add Connection



(c) Multiply Connection

Fig. 21: Comparison of Concatenation, Add, and Multiply Connections at 200 Epochs (learning rate = 0.0002, Batch Size = 1)