CS230

Generating Images of the Consequences of Wildfire

Minh Nguyen
Department of Computer Science
Stanford University
mnguy@stanford.edu

Chang Wan Ryu
Department of Computer Science
Stanford University
galmacky@stanford.edu

David Wang
Department of Computer Science
Stanford University
daviddw@stanford.edu

Abstract

We present a project that aims to generate realistic images of houses on fire, to visualize the devastating consequences of wildfires. We experimented with several different architectures and approaches: Cycle-Consistent Adversarial Network (CycleGAN), Contrastive-Unpaired-Translation (CUT), and Multimodal Unsupervised Image-to-image Translation (MUNIT). In our results, CycleGAN best preserved building realism while MUNIT generated the most realistic fire and smoke.

1 Introduction

Wildfires displace populations, destroy property, and produce harmful pollutants which pose serious health risks. Their aftereffects include erosion, debris flows, and altered water quality. The impact of climate change is reflected in the unprecedented number of wildfires in California this year (1). This project aims to generate realistic images visualizing the devastating consequences of wildfires. The results of this project may be used to raise awareness for the impacts of climate change and potentially to aid in the creation of fire-based special effects.

2 Related Work

Style transfer has been applied across problem domains ranging from the relatively unconstrained case of art generation to problems requiring photo-realism such as weather manipulation. The most closely related work to this project is a pair of papers presenting the generation of flooded street-view images from non-flooded images by Victor Schmidt et al in 2019 (2) and by Gautier Cosne et al in 2020 (3).

Architecture literature review included a broad review of popular image-to-image translation networks, with a focus on state of the art architectures for image translation including CycleGAN by Jun-Yan Zhu et al (4), CUT by Taesung Park et al (5), and MUNIT by Xun Huang et al (6). The effectiveness of the chosen architecture hinged on its ability to generate realistic fire textures and maintain image content (house, street) while modifying style (burning environment).

CS230: Deep Learning, Fall 2020, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

3 Dataset and Features

The unaugmented primary training dataset consists of approximately 400 street-view images of buildings on fire and approximately 400 street-view images of regular buildings, each with at least 512x512 resolution. Additional lower resolution images were used as supplemental data when training models with lower input resolution. The training dataset images were fetched from multiple sources. Images of both classes were collected using a web scraper that queried Google image search, with downloaded images curated to remove any image that was not a building nor a building on fire. Additional images of houses on fire were collected using Flickr and by extracting frames from GIFs and videos. We applied data augmentation techniques such as horizontal flipping and small rotations with random cropping, but decided to only augment through image flipping due to training time constraints. All images in our training dataset are unpaired.

The test dataset consists of 11 paired images of the same location before and during a wild-fire and were collected through manual search. This allowed us to directly compare the output of our model with the actual appearance of the building on fire.

4 Methods

We experimented with several GAN models that represent the current state of the art in unsupervised image-to-image translation. This task is considered unsupervised because models train on unpaired image data, which is necessary given the lack of paired image data in the problem domain. We experimented with various input image resolutions so some models were trained only using relatively high resolution images (at least 512x512) and some with all the images we collected.

4.1 Cycle-Consistent Adversarial Network (CycleGAN)

4.1.1 Description

CycleGAN (4) trains two pairs of generator and discriminator models to learn both a mapping for generating an image and its inverse. For two image domains X and Y, a house not on fire and a house on fire for our task, CycleGAN trains a generator that learns $G_1 : X \rightarrow Y$ and a generator that learns $G_2 : Y \rightarrow X$. Cycle consistency is added to the loss function to enforce $G_2(G_1(X)) \approx X$. Loss functions are in Appendix A.

4.1.2 Application and Hyperparameters

We modified CycleGAN code from the Tensorflow Core tutorial for CycleGAN (7) to use our dataset as input. The generator and discriminator are imported from Pix2pix but instance normalization was used instead of batch normalization for this application. The generator also uses a Unet architecture and not a ResNet arechitecture. We used default parameters and trained for 400 epochs, which is longer than the original paper (200) and provided noticeably better results. Input images were rescaled to fit model specifications and we trained on both 256x256 resolution images as well as higher resolution 512x512 images.

4.2 Contrastive-Unpaired-Translation (CUT)

4.2.1 Description

Contrastive-Unpaired-Translation (5) maximizes mutual information between the patch in the input and the patch in the output, using a framework based on contrastive learning. Compared to CycleGAN, CUT learns to perform more powerful distribution matching. Loss functions are in Appendix A.

4.2.2 Application and Hyperparameters

The CUT Github repository linked in Park et al (5) was forked into a separate project repository for customization. CUT was generally slower than other approaches due to complexity around the patchwise application of contrastive learning. The initial model was trained with anti-aliased downsampling and upscaling using a hard-coded filter, but results showed a grid of blurriness.

Blurriness was identified as the primary deficiency of our initial CUT model, so the final model was trained without this downsampling and upscaling. This approach improved visual clarity but each training step took four times longer. Therefore, we had to stop training half way through the default 400 epochs. The final model was trained on 256x256 input images for 150 epochs over a duration of 72 hours on an AWS instance.

4.3 Multimodal UNsupervised Image-to-image Translation (MUNIT)

4.3.1 Description

MUNIT (6) assumes that each image can be decomposed into a domain-invariant content code and a domain-specific style code. To translate an image to another domain, its content code is combined with a style code sampled from the target domain's style space. MUNIT learns a multimodal conditional distribution of possible outputs for each input image, rather than a deterministic one-to-one mapping like CycleGAN and CUT. Loss functions are in Appendix A.

4.3.2 Application and Hyperparameters

The NVlabs/MUNIT Github repository linked in Huang et al (6) was forked into a separate project repository for customization. We wrote a Colab script to install dependencies, preprocess images, and run model training and testing on a remote machine. The Yosemite summer-to-winter config file provided in the repository was used as a starting point for our training configuration based on content similarity (e.g. outdoor environment, vegetation). The initial model was trained on 256x256 input images, and the primary deficiencies were determined to be distortion and blurriness. Higher resolution input images were used in an attempt to mitigate these issues. The final model was trained at a resolution of 360x360 with a batch size of one on a 12GB NVIDIA Tesla K80 GPU, reaching a total of 100,000 iterations. Total training time was increased from 30 hours to 80 hours between the initial model and final model to account for the increase in input image resolution and dataset size. Training on resolutions higher than 360x360 exceeded GPU memory.

5 Results and Discussion

5.1 CycleGAN Results and Analysis

Figure 1: Images generated from CycleGAN generator

Increasing the resolution of the input and output images from 256x256 to 512x512 improved the realism and quality of fire, smoke, and houses in the generated images. While preserving the appearance of buildings, CycleGAN was able to generate some images with realistic flames composed of gradients of orange and red and realistic smoke composed of gradients of grey and black. However, sometimes the generated images would clumsily tint regions of the image orange or black or simply darken the image to make it look like night time.

Looking at CycleGAN's loss graph (Appendix D), cycle consistency loss was the largest loss, so the model may have focused too much on cycle consistency as reflected in its superior ability to maintain building realism and mediocre generation of fire and smoke.

5.2 CUT Results and Analysis

Figure 2: Images generated from CUT generator

Compared to CycleGAN at the same training resolution (256x256), our final CUT model showed comparable overall results. Fire and smoke were in reasonable locations, but the actual appearance of flames and smoke was not realistic. Removing anti-aliased downsampling and upscaling reduced blurriness compared to our initial model but significantly increased training time. In the interest of maintaining comparable training time between models, as well as general time constraints, final model training was stopped at 72 hours, which was less than half way through the default 400 epochs. We believe additional training would have improved fire and smoke realism, as our initial model was assessed to have more crisp and realistic flames than CycleGAN. Thus we note training time as a downside of CUT compared to CycleGAN.

5.3 MUNIT Results and Analysis

Figure 3: Images generated from MUNIT generator

Increasing input image resolution noticeably reduced blurriness and overall distortion. MUNIT was better able to capture style characteristics of flames and smoke compared to other models but often distorts the appearance of image content by applying unrealistic color shifts or adding noisy artifacts. This may be a result of applying styles to poorly mapped content codes. We hypothesize that MUNIT requires additional input data of both classes to adequately map the content space, as distortion occurs most often when uncommon objects appear in input images (e.g. oddly colored first stories of buildings, uncommon building facades, large bodies of water). Our dataset is both smaller and more varied than the flooding dataset used by Cosne et al, which consisted of 2000 unpaired real images and 2000 paired simulated images, all curated to minimize extraneous objects (3). Additionally, MUNIT does not penalize distortion as heavily as CycleGAN and CUT by default, which highly weigh cycle-consistency loss and patchwise contrastive loss respectively.

5.4 Comparison Between Methods

Input CycleGAN MUNIT CUT Real Fire

Figure 4: Side-by-side comparison of images generated based on a single test image, other test images displayed in Appendix B

The members of our group completed a survey of generated image quality based on three categories: fire, smoke, and building realism. Using a 5-point rating system, the results were as follows:

Method	Fire Realism	Smoke Realism	Building Realism	Avg
CycleGAN 256x256	1.76	2.03	2.42	2.07
CycleGAN 512x512	1.81	2.48	**4.27**	**2.86**
CUT 256x256	1.58	2.15	2.64	2.12
MUNIT 360x360	**2.27**	**2.76**	1.94	2.32

Comparing our scores, CycleGAN using 512x512 images was the best at preserving building realism while MUNIT was the best at generating realistic looking fire and smoke. Detailed scoring in Appendix C.

6 Conclusion

The CycleGAN model trained on 512x512 images had the highest overall performance due mainly to a superior ability to preserve input image details. MUNIT was best able to capture the style characteristics of fire and smoke, but often distorted the appearance of image content by applying unrealistic color shifts or adding noisy artifacts. CUT didn't progress as far in training as the other models after removing anti-aliased downsampling and upscaling, which likely explains its lower fire and smoke realism.

7 Future Work

The CycleGAN model could be improved by changing the weights within the loss function, specifically reducing cycle consistency loss weight to encourage more realistic fire and smoke generation. MUNIT could be improved by adding a mask to restrict changes to the foreground, specifically around buildings, trees, and other highly flammable objects. This would help limit the content space that MUNIT needs to learn, reducing the likelihood of applying styles to poorly mapped content codes. The main bottleneck for our final CUT model was training time so training longer would likely improve model performance. Additional possible future work includes creating a larger and more strictly curated dataset as well as experimenting with additional GAN architectures.

8 Contributions

Changwan and Minh used a Google image search scraper to download images of normal houses and houses on fire for the dataset. David extracted images from Flickr, GIFs, and videos for the datasets. All members helped with curating the dataset. Each approach was covered by a different member: Minh trained and ran CycleGAN, Changwan trained and ran CUT, and David trained and ran MUNIT.

References

[1] A. Freedman, H. Kelly, H. Knowles, and J. Whalen, *California wildfires reach historic scale and are still growing*, 2020. `https://www.washingtonpost.com/weather/2020/08/22/california-wildfires-largest/`.

[2] V. Schmidt, A. Luccioni, S. K. Mukkavilli, N. Balasooriya, K. Sankaran, J. Chayes, and Y. Bengio, "Visualizing the consequences of climate change using cycle-consistent adversarial networks," 2019.

[3] G. Cosne, A. Juraver, M. Teng, V. Schmidt, V. Vardanyan, A. Luccioni, and Y. Bengio, "Using simulated data to generate images of climate change," 2020.

[4] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," 2020.

[5] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, "Contrastive learning for unpaired image-to-image translation," 2020.

[6] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," 2018.

[7] *CycleGAN*, 2020. `https://www.tensorflow.org/tutorials/generative/cyclegan`.

9 Appendix A: Loss Functions of Models

9.1 CycleGAN Loss Functions

GAN loss, cycle consistency loss, and total loss are defined as follows:

$$\mathcal{L}_{GAN}(G, D, X, Y) = \mathbb{E}_{y \sim p_{data}(y)} log(D(y)) + \mathbb{E}_{x \sim p_{data}(x)} log(1 - D(G(x))) \tag{1}$$

$$\mathcal{L}_{cyc}(G_1, G_2) = \mathbb{E}_{x \sim p_{data}(x)} ||G_2(G_1(x)) - x||_1 + \mathbb{E}_{y \sim p_{data}(y)} ||G_1(G_2(y)) - y||_1 \tag{2}$$

$$\mathcal{L}_{total}(G_1, G_2, D_1, D_2) = \mathcal{L}_{GAN}(G_1, D_1, X, Y) + \mathcal{L}_{GAN}(G_2, D_2, Y, X) + \lambda \mathcal{L}_{cyc}(G_1, G_2) \tag{3}$$

9.2 CUT Loss Functions

GAN loss, patchwise contrastive loss, multi-layer patchwise contrastive loss, and total loss are defined as follows:

$$\mathcal{L}_{GAN}(G, D, X, Y) = \mathop{\mathbb{E}}_{y \sim Y} \log D(y) + \mathop{\mathbb{E}}_{x \sim X} \log(1 - D(G(x)) \tag{4}$$

$$\ell(v, v^+, v^-) = -\log[\frac{\exp(v \cdot v^+ / \tau)}{\exp(v \cdot v^+ / \tau) + \Sigma_{n=1}^{N} \exp(v \cdot v_n^- / \tau)}] \tag{5}$$

$$\mathcal{L}_{PatchNCE}(G, H, X) = \mathop{\mathbb{E}}_{x \sim X} \Sigma_{l=1}^{L} \Sigma_{s=1}^{S_l} \ell(\hat{z}_l^s, z_l^s, z_l^{S \setminus s}) \tag{6}$$

$$\mathcal{L}_{GAN}(G, D, X, Y) + \lambda_X \mathcal{L}_{PatchNCE}(G, H, X) + \lambda_Y \mathcal{L}_{PatchNCE}(G, H, Y) \tag{7}$$

9.3 MUNIT Loss Functions

Reconstruction loss, adversarial loss, and total loss are defined as follows:

$$\mathcal{L}_{recon}^{x_1} = \mathbb{E}_{x_1\ p(x_1)}||G_1(\mathbb{E}_1^c(x_1), \mathbb{E}_1^s(x_1)) - x_1||_1 \tag{8}$$

$$\mathcal{L}_{recon}^{c_1} = \mathbb{E}_{c_1\ p(c_1), s_2\ q(s_2)}||\mathbb{E}_2^c(G_2(c_1, s_2)) - c_1||_1 \tag{9}$$

$$\mathcal{L}_{recon}^{s_2} = \mathbb{E}_{c_1\ p(c_1), s_2\ q(s_2)}||\mathbb{E}_2^s(G_2(c_1, s_2)) - s_2||_1 \tag{10}$$

$$\mathcal{L}_{GAN}^{x_2} = \mathbb{E}_{c_1\ p(c_1), s_2\ q(s_2)} log(1 - D_2(G_2(c_1, s_2))) + \mathbb{E}_{x_2\ p(x_2)} log(D_2(x_2)) \tag{11}$$

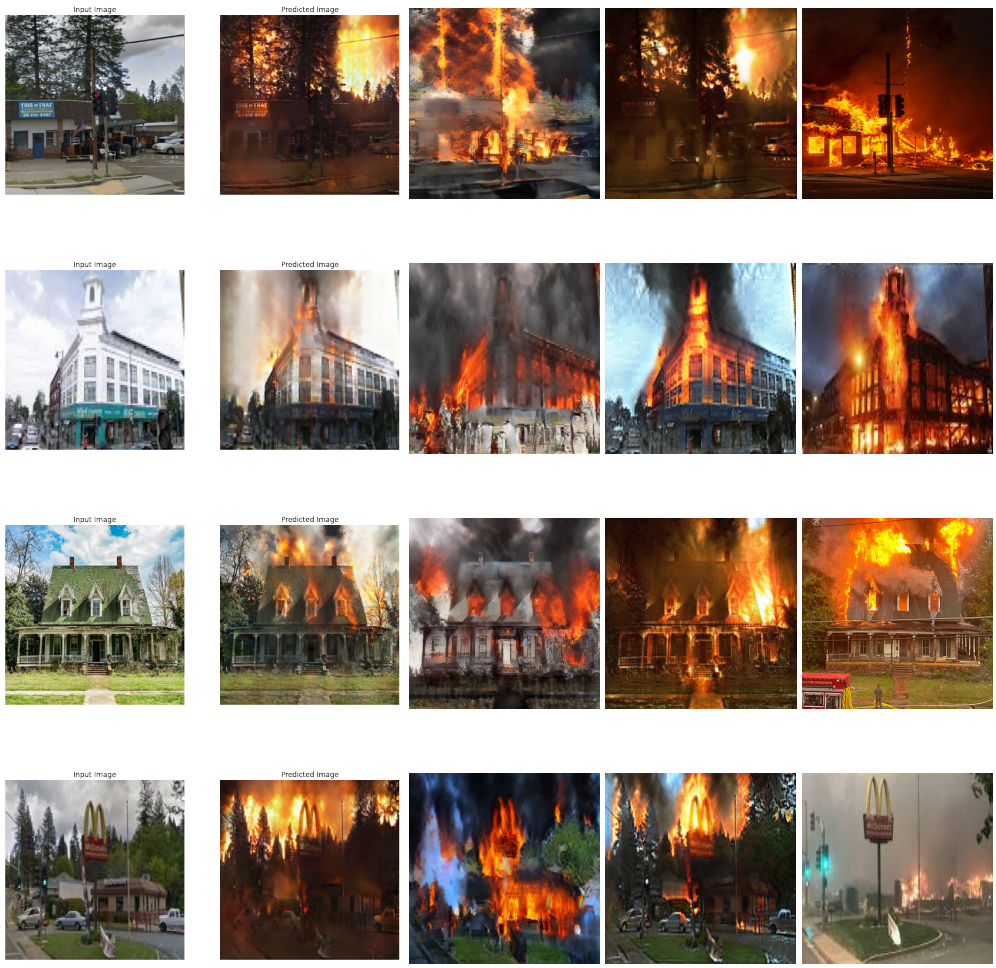
$$\min_{E_1, E_2, G_1, G_2} \max_{D_1, D_2} \mathcal{L}_{total}(E_1, E_2, G_1, G_2, D_1, D_2) = \mathcal{L}_{GAN}^{x_1} + \mathcal{L}_{GAN}^{x_2} + \lambda_x(\mathcal{L}_{recon}^{x_1} + \mathcal{L}_{recon}^{x_2}) + \lambda_c(\mathcal{L}_{recon}^{c_1} + \mathcal{L}_{recon}^{c_2}) + \lambda_s(\mathcal{L}_{recon}^{s_1} + \mathcal{L}_{recon}^{s_2}) \tag{12}$$

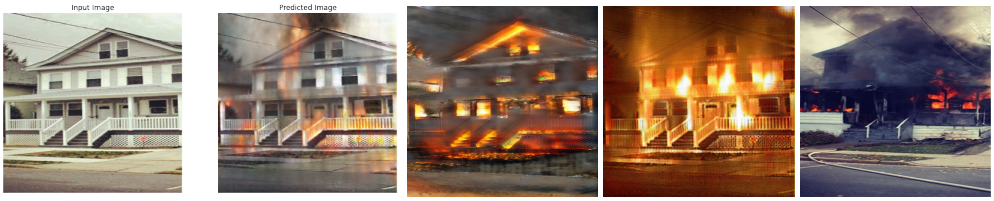
Where $L_{recon}^{x_2}$, $L_{recon}^{c_2}$, $L_{recon}^{s_1}$, and $L_{GAN}^{x_1}$ are defined in a similar manner to their counterparts and the translation model consists of two auto-encoders.

10 Appendix B: Generated Images using Test Set

We compared the outputs of the three algorithms against 11 paired images of before and during fire.

Input | CycleGAN | MUNIT | CUT | Real Fire




Input Image
Predicted Image


Input Image
Predicted Image


Input Image
Predicted Image


Input Image
Predicted Image


Input Image
Predicted Image


Input Image
Predicted Image

11 Appendix C: Survey of Quality of Generated Images

CycleGAN 256x256 score	Fire Realism	Smoke Realism	Building Realism
Evaluator A's avg score	2.091	1.909	2.272
Evaluator B's avg score	1.727	2.273	2.455
Evaluator C's avg score	1.455	1.909	2.545
Total avg	1.758	2.030	2.424

CycleGAN 512x512 score	Fire Realism	Smoke Realism	Building Realism
Evaluator A's avg score	2.273	2.727	4.727
Evaluator B's avg score	1.727	2.636	4.182
Evaluator C's avg score	1.455	2.091	3.909
Total avg	1.818	2.485	4.273

CUT 256x256 score	Fire Realism	Smoke Realism	Building Realism
Evaluator A's avg score	1.727	2.455	2.727
Evaluator B's avg score	1.454	2.091	2.636
Evaluator C's avg score	1.545	1.909	2.545
Total avg	1.576	2.151	2.636

MUNIT 360x360 score	Fire Realism	Smoke Realism	Building Realism
Evaluator A's avg score	2.364	3.000	2.091
Evaluator B's avg score	2.273	2.727	1.818
Evaluator C's avg score	2.182	2.545	1.909
Total avg	2.273	2.758	1.939

12 Appendix D: Training Loss Graphs

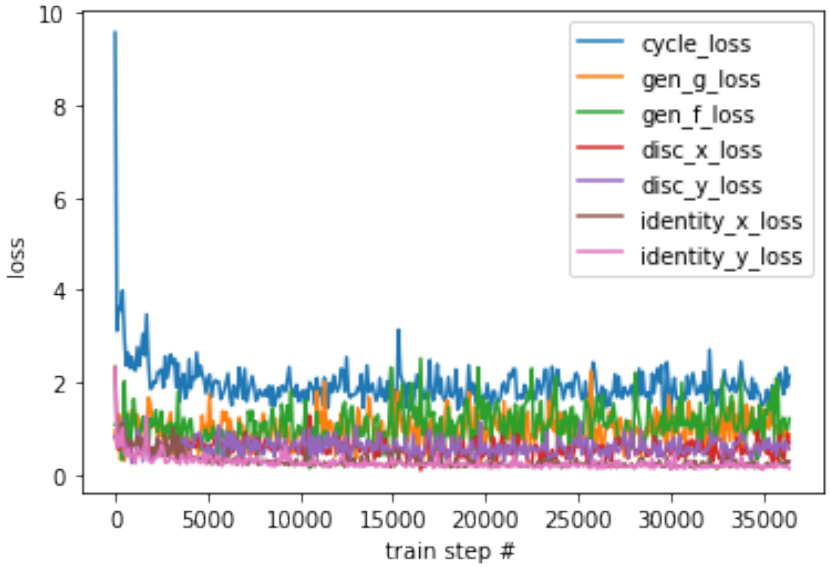


Figure 5: CycleGAN Loss Graph

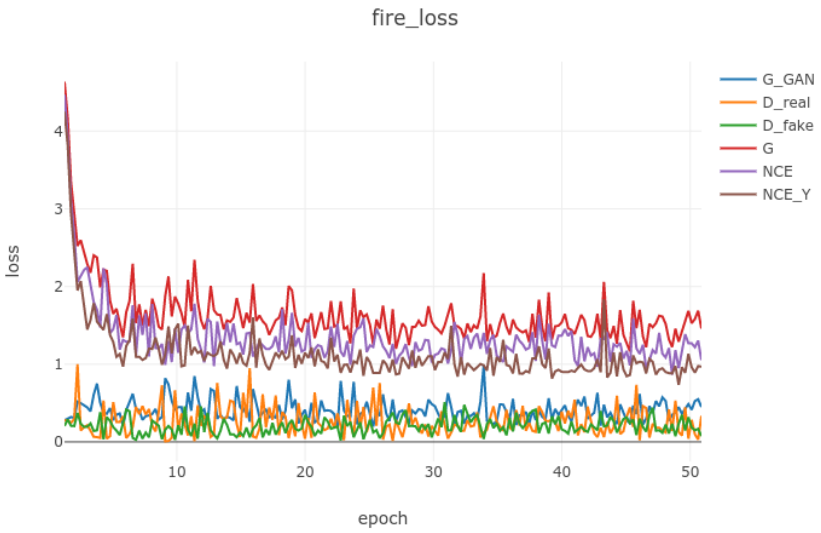


Figure 6: CUT Loss Graph

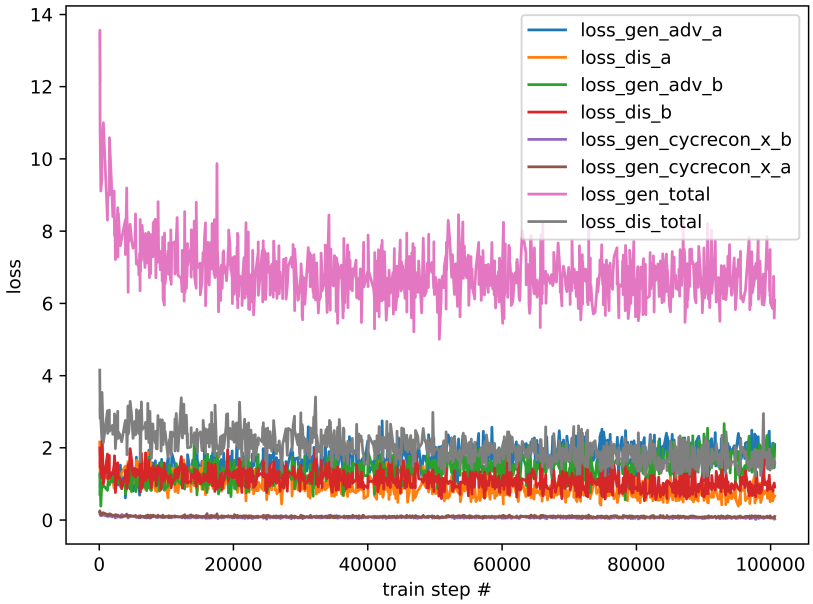


Figure 7: MUNIT Loss Graph