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

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 Dataset and Data Collection

The training dataset consists of around 250 street-view images of buildings on fire and around 250 street-view images of regular buildings. The training dataset image was generated by using a web scraper that queried Google image search and downloaded many images from the results. The downloaded images were curated to remove any image that was not a building nor a building on fire. We used basic data augmentation techniques to generate additional data, including horizontal flipping, cropping, and adding noise. All images in our training dataset are unpaired.

The test dataset consists of paired images of the same location before and during a wildfire and were found by manually searching. With this, we can directly compare the output of our model with how the building looked during an actual wildfire.

3 Approach

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.

3.1 CycleGAN

CycleGAN (2) 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 on not on fire and on fire for our task, CycleGAN trains a generator that learns $G_1 : X \rightarrow Y$ and a generator that learns

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

$G_2 : Y \rightarrow X$. Cycle consistency is added to the loss function to enforce $G_2(G_1(X)) \approx X$. GAN loss, cycle consistency loss, and total loss are defined as follows:

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

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

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

We trained the CycleGAN model on our training dataset for 250 epochs, around 15 hours, using GPU. We used the generator model that mapped from normal buildings to buildings on fire to generate our output.

3.2 Contrastive-Unpaired-Translation (CUT)

Contrastive-Unpaired-Translation(3) 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. GAN loss, patchwise contrastive loss, multi-layer patchwise contrastive loss, and total loss are defined as follows:

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

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

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

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

We trained CUT for 200 epochs (24 hours on AWS instance).

3.3 Multimodel UNsupervised Image-to-image Translation (MUNIT)

MUNIT (4) 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. 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 consisting of two auto-encoders is shown in Appendix D. The MUNIT model was trained for approximately 30 hours on a Tesla K80 GPU.

4 Preliminary Results and Analysis

We completed a survey of quality among ourselves for the generated images based on three categories: fire, smoke, and building, using a 5-star rating system, and got the following results:

Method	Fire Realism	Smoke Realism	Building Realism	Avg
CycleGAN	2.09	**2.58**	**3.30**	**2.66**
MUNIT	**2.61**	2.36	1.70	2.22
CUT	**2.61**	2.45	2.76	2.61

Comparing our scores, CycleGAN was noticeably worse at generating fire than both MUNIT and CUT, however CycleGAN retained the building better than both MUNIT and CUT. This is likely due to low output resolution of both MUNIT and CUT models. All models performed similarly for generating smoke, with CycleGAN performing the best.

The generated images and survey details can be found in Appendix A and B.

4.1 CycleGAN

CycleGAN was able to capture the coloration of fire and the environment (i.e. smoke and nighttime) and applies the fire coloring to reasonable parts of the image: on the sides of buildings usually near windows and on the upper floors. The main problems with the outputs of CycleGAN are that flames don't look very realistic, fire coloration is sometimes applied to the wrong parts of the image (i.e. background), and the generator is heavily biased towards making images in night time.

4.2 MUNIT

A specific style sampling strategy was chosen to avoid giving MUNIT an advantage through manual curation. MUNIT was better able to capture flame characteristics compared to other models. However, generated images suffered from poor resolution and a significant amount of distortion. This may be due to the higher computational cost of training a two auto-encoder model, resulting in slower model improvement. Results may benefit from training on higher resolution images with more powerful GPUs and a longer duration. For comparison, Yosemite summer to winter image translation presented by Huang et al. (2018) was trained using 1024x1024 resolution images on an NVIDIA DGX1 with 8 V100 32GB GPUs (4).

4.3 CUT

Compared to CycleGAN, the rendering of flames was more crisp and more realistic. Smoke was also seen as decently realistic. However, in our experiment, every output image had blurriness that negatively affected image quality.

5 Future Work

We intend to collect a greater number of images to augment the size of our dataset. GAN models for image generation are frequently trained on thousands of images in literature. Cosne et al. (2020) built a virtual city for the purpose of augmenting their dataset (5). While such an approach would be infeasible given project time constraints, additional data augmentation techniques (e.g. color shift) can be used to augment our dataset. All models were trained on images preprocessed to a resolution of 256x256. We intend to increase image resolution for training and testing due to our suspicion that training on low-resolution images contributed to blurriness issues in MUNIT and CUT. We can also add a mask to isolate fire and smoke generation to a limited area around the building, and not other parts of the image such as the background. Lastly, we can try tweaking the loss function and tuning the hyperparameters in each approach to see if the flames can be generated more realistically. We could also search for a pre-trained model and do transfer learning.

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

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

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

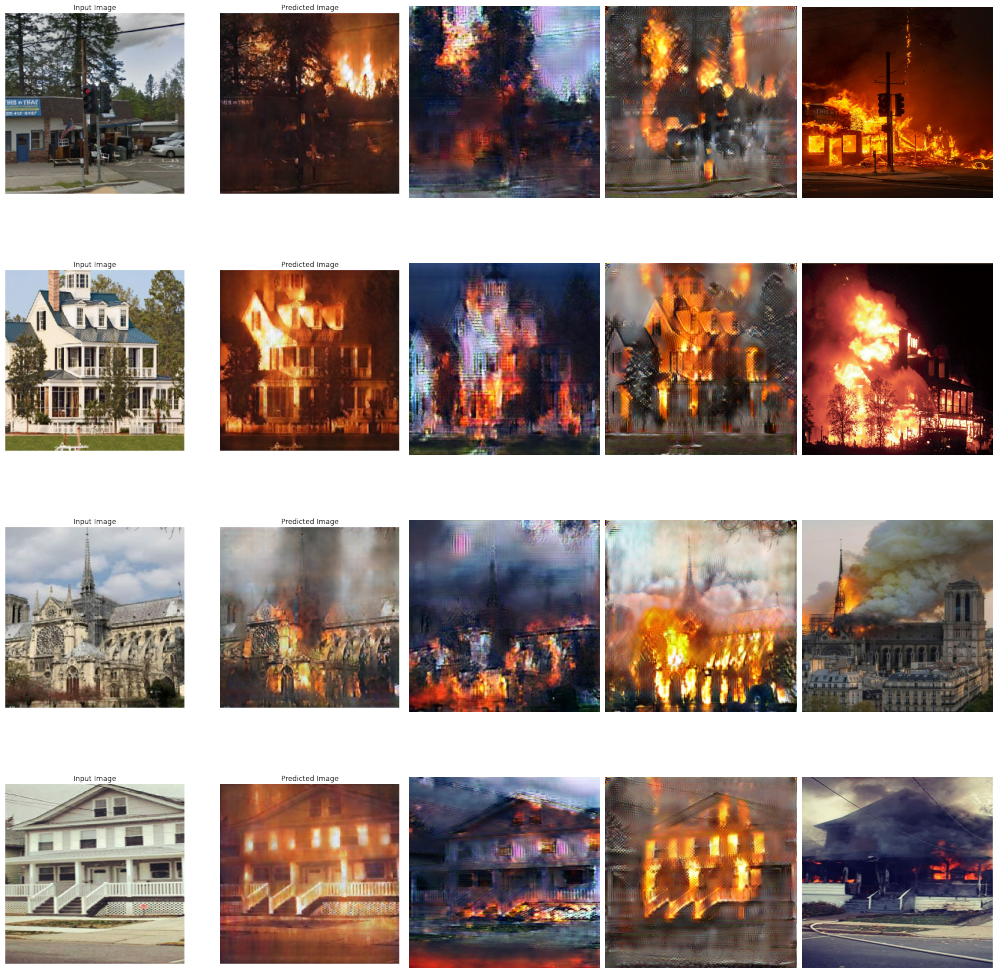
[5] G. Cosne, A. Juraver, M. Teng, V. Schmidt, V. Vardanyan, A. Luccioni, and Y. Bengio, "Using simulated data to generate images of climate change," 2020.

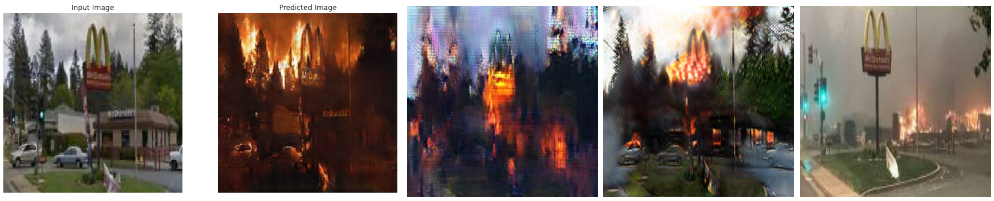
[6] G. Ghiasi, H. Lee, M. Kudlur, V. Dumoulin, and J. Shlens, "Exploring the structure of a real-time, arbitrary neural artistic stylization network," 2017.

6 Appendix A: Generated Images

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
Input Image
Predicted Image

7 Appendix B: Survey of Quality of the Generated Images

CycleGAN score	Fire Realism	Smoke Realism	Building Realism
Evaluator A's avg score	2.091	3.000	3.545
Evaluator B's avg score	2.000	2.545	3.000
Evaluator C's avg score	2.182	2.182	3.364
Total avg	2.091	2.576	3.303

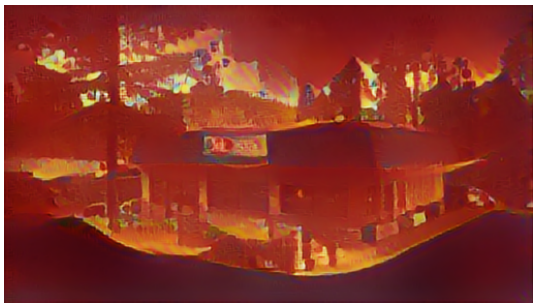
MUNIT score	Fire Realism	Smoke Realism	Building Realism
Evaluator A's avg score	2.455	2.273	1.727
Evaluator B's avg score	2.636	2.364	1.636
Evaluator C's avg score	2.727	2.455	1.727
Total avg	2.606	2.364	1.697

CUT score	Fire Realism	Smoke Realism	Building Realism
Evaluator A's avg score	2.818	2.818	2.545
Evaluator B's avg score	2.454	2.182	2.727
Evaluator C's avg score	2.545	2.364	3.000
Total avg	2.606	2.455	2.756

8 Appendix C: Style Transfer

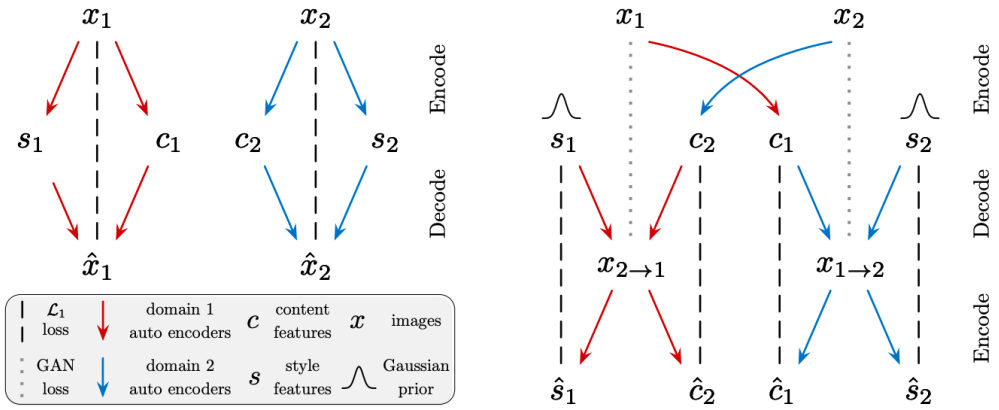
A discarded approach was to use style transfer with the the content image being a normal building and the style image being a building on fire. Style transfer is a computer vision technique to generate a new image that contains the "content" of one image and the "style" of another. We used Fast Style Transfer developed by Ghiasi et al (6) to generate the image.

We did not get a meaningful result from Style Transfer. Using a wildfire image as the style image added a lot of orange and black coloring to the content image. While the output makes sense, but the image isn't realistic and doesn't resemble real fire. This approach will be discarded in favor of GAN approaches.



9 Appendix D: MUNIT image-to-image translation model

The image-to-image translation model consists of two auto-encoders (one for each domain). Each auto-encoder's latent code is composed of a content code c and a style code s. Adversarial objectives (dotted lines) ensure that translated images are indistinguishable from real images in the target domain. Bidirectional reconstruction objectives (dashed lines) reconstruct both images and latent codes (4).



(a) Within-domain reconstruction

(b) Cross-domain translation

Figure 1: MUNIT image-to-image translation model (4)

10 Appendix E: Github Link

http://github.com/galmacky/env_gan