Personal Climate Change

Roei Ben Zion, Dima Kisileiv, Daniella Glozman

Computer Science department, Tel Aviv University

GitHub

Figure 1: the 3 climate change effects we presented, the first (upper left) simulates flood in a street image, the second (upper right) simulates stormy weather in a sunny weathered image and the third (lower middle) simulates extreme heatwave.

Abstract

In this work, we aim to take advantage of the recent advancements in the fields of image generation and translation to build a tool that could help raise awareness to the climate change problem among the public, by simulating 3 selected effects of climate change on an input image. For each damage, we use a different method that in our view fits the best by both performance and resources needed to train it. For heatwaves, we use the classic CycleGAN [2] to learn a mapping to simulate the input image under the effect of an extreme heatwave. For floods, CycleGAN by itself yielded not sufficient results, so we adopt the InstaGAN [3] architecture and loss function, which is based on CycleGAN but specializes in translation of some instance in the image and preserving the rest of it. Finally, for stormy weather simulation, we use Neural Style Transfer [4]. This method is much less resource demanding and gives us an opportunity to experiment with another technology.

Introduction

Problem statement – In order to raise awareness to the climate change problem in public we adopt a somewhat straightforward approach, based on a cold assumption that people have a hard time relating to something that appears far from them. Hence, our goal is to show the user how some effects that are already happening today would look like in their own natural environment, hoping this will provide a better motivator to act. We focus on imagery, since we think it is the most powerful way to demonstrate the effects.

Approach – We aimed to take advantage of the popular GAN [1] architecture and its variants because of its great performance in the image generation and translation fields and relatively well-established literature and information sources. We've chosen 3 damages of climate change – flooding, heatwaves and stormy weather, these choices are based on recent world events as well as the existence of accessible resources (e.g., data). Aiming for perfect personalization, we focused on the Image-to-Image translation technologies, and more specifically unpaired Image-to-Image translation (I2I translation). In I2I translation we have 2 domains of images, A and B, and we are trying to find a meaningful mapping $G_A: A \rightarrow B$ while preserving the content

representations. In the paired I2I translation we are adding the assumption that every image $x \in A$ has a corresponding image $y \in B$ such that a meaningful translation exits between the 2. However, in most real-life applications, and especially in ours, it is not realistic to make this assumption, so we will work in an unpaired setting. For every effect, we will use domain A (source) as the domain of images without this effect and domain B (target) would be the images with this effect.

Methods

In the following we describe the methods we used for every damage separately, since we used different methods for each to get the best results with our limited resources. As there is a lack of clear consensus regarding reliable evaluation metrics for CycleGAN, in both heatwaves and floods we used user studies and empirical evaluation as evaluation metrics on the test set. We'll note that as far as we know there are no prepared dataset for the following tasks, so we had to build (collect) them ourselves.

1. Heatwaves - Heatwaves have been a real-world problem in recent years, high temperatures are breaking records even in some areas where heat was not a serious problem before such as north Europe a and more. We aim to show visually how an extreme heatwave could have looked like even in some areas that haven't experienced one yet. To achieve that, we set domain A to be images of locations with no visible heatwave, and domain B images of landscape of locations struck by a visible heatwave, we use the classic CycleGAN for the task. In the CycleGAN setting we train 2 generators and 2 discriminators G_A, G_B, D_A, D_B using some version of the GAN adversarial loss, and adding 2 constrains (for both sides) to get a meaningful mapping:
1 – the cyclic loss $L_{cyc} = ||G_B(G_A(x)) - x||_1$

2 – the identity loss $L_{idt} = \left||G_B(y) - y|\right|_1$

When $G_A: A \to B, G_B: B \to A, x \in A, y \in B$. We have adopted the Unet-based generator architecture introduced in the official implementation since it yielded much better results than the Resnet-based one. To choose the proper adversarial loss, we've experimented with 3 popular losses, the default least-squares loss, the hinge GAN loss, and the Wasserstein GAN (WGAN) loss [5]. We have also examined 3 regularization factors: Total Variation (TV) denoiser [6] perceptual and style as in [3] . We have found that the best combination is the WGAN loss when the discriminators parameters are clipped in $[-0.01, 0.1]$ and the TV denoiser, where we aim to minimize the difference between generated image pixels, hence our final loss function was:

$$L_{heat} = \lambda_1 L_{WGAN} + \lambda_2 L_{cyc} + \lambda_3 L_{idt} + \lambda_4 L_{TV}$$

When (assuming z is a generated image):

$$L_{WGAN} = E_{x \sim P_A}[D_A(x)] - E_{y \sim P_B}[D_A(G_B(y)] + E_{y \sim P_B}[D_B(y)] - E_{x \sim P_A}[D_B(G_A(x)]$$

$$L_{TV} = \sum_{i=1}^{H} \sum_{j=1}^{W-1} |z_{i,j} - z_{i,j+1}| + \sum_{i=1}^{H-1} \sum_{j=1}^{W} |z_{i,j} - z_{i+1,j}|$$

$$\lambda_1 = \lambda_2 = 10,\ \lambda_3 = 0.5,\ \lambda_4 = 10^{-5}$$

Figure 2: selected results of our model, the daylight images are the input.

We run the model with L_{heat} for 100 epochs on a dataset of around 200 images per domain that we've collected from the web.

2. Floods – out of the 3, floods simulation was the most complex task, and we've seen that CycleGAN (and some of its variants) fails to learn this translation. To solve this problem, we have decided to use a model named InstaGAN, which showed promising results in an instance aware translation, when the task is to translate some part of the image and preserve the rest. Differently from CycleGAN, the mapping InstaGAN aims to learn is from a paired domain of images and instances that are masked regions in each image to a similar structured domain, when the goal is to change only the generated masked part. In other words, we aim to learn G_A: $A\ X\ I_A \rightarrow\ B\ X\ I_B$ (where the instance in I_A is the road and sidewalk of every image) and the same for B (when the instance is the flooded part). To build such a dataset for the flooding task we need to be able to mask the flooded (water) part in domain B, and some part in the images of domain A we would want to see flooded (the road and sidewalk).

2.1 Segmentation – In order to be able to train an InstaGAN model, we had to create domains of images and corresponding masks. For the domain of images with floods (B), we masked the area where there's water, in domain A we've decided to segment roads, sidewalks etc. For domain A we used the famous CitySpaces [7] dataset that fortunately comes with a full semantic segmentation for every image. For the flood's domain, we have tried to build our own masking model to mask the water in an image, using Unet architecture and the Atlantis [12] dataset. We have compared this model to CSAIL segmentation model [8], and the latter showed better results, so we sticked with it for building our dataset's domain B.

2.2 Data preprocessing – The floods dataset, FloodIMG [9], contains many images that could not be assumed as corresponding to CitySpaces by some meaningful mapping, such as rivers and aerial images, and on the other hand it is quite large (more than 5000 images), and one cannot extract selected images by hand.

We've tried 2 different ways to tackle that problem and trained the model with each, although the end results were similar in both. In the first way we used a Resnet backbone (by removing last layer of Resnet18) pre-trained on ImageNet to extract semantic features of every flooded image and of a sample of 150 randomly selected images from the CitySpaces dataset, then we average the 150 images resulting feature map. We then used the cosine similarity between each flooded image's feature map to the average and chose the top 1500 flood images.

The second way was to build a classifier to classify between images of street flood (desired) and the rest (river, aerial and more undesired). We've trained this using 200 images in training set (100 per domain) when they are all taken and labeled by hand from the FloodIMG dataset. We've built it by fine tuning a pre-trained VGG16 net. We run the classifier on the FloodIMG dataset and end with 1592 relevant images for domain B. The model we present here is the one trained on the dataset the classifier created.

2.3 Training – We train an InstaGAN model using a Resnet-based generator, and a least-squares adversarial loss for the GAN. Differently from the method used in the InstaGAN paper, we let the weight of the identity loss and the context loss to be 1/10 from the adversarial loss and the cycle loss in all our experiments as other configurations we tried yielded unstable results.

We have also looked for ways to improve the performance of the model. Following one method proposed in [10], we smooth the discriminator target in the adversarial loss function, so that instead of targets of 1 and 0, we generate a random label $target \sim Unif((0,1))$ multiply it by 0.1 and adding 0.9 iff the input is real. In addition, we experiment with adding the Convolutional Block Attention Module (CBAM) [11] to the Resnet blocks in the generator, as it showed promising results in image classification and object detection tasks and examine its impact in image generation task.

The best way we have found to add CBAM to the Resnet blocks in our generator is the following: we take the Resnet block's output and pass it through a channel attention module and then through a spatial attention module (in the same structure as the paper presents). Then we add a residual connection between the original Resnet block output and the output after the attention modules, finally we add non-linearity (ReLu). We

have witnessed that combining label smoothing and CBAM yields in highly unstable results from the generator with noisy outputs, and that label smoothing on its own does not change the results.

However, adding CBAM to the Resnet blocks as we did have changed the training significantly. The model with CBAM seems to add the flood in a more stable fashion way faster than the baseline model, but several epochs afterwards it starts losing its ability.

Figure 3: training process in epochs 8,15,30 (left to right) of the baseline model (upper) in comparison to the CBAM model (lower) on the classifier generated dataset. We can see that CBAM is much better in epoch 8, and almost reaches a good result as soon as epoch 15, when it has almost no change outside the flooded part and the baseline model is still noisy there. However, in epoch 30 the roles are turned, and the baseline model comes on top.

We've also noticed that the CBAM model, until it collapses, is never noisy outside of the changed part, in contrast to the baseline. That being said, it still collapsed in higher epochs with all hyperparameter settings we've tried.

Our final loss function in the baseline model is:

$$L_{floods} = \lambda_1 L_{LS-GA} \ + \lambda_2 L_{cyc} + \lambda_3 L_{idt} + \lambda_4 L_{ctx}$$

$$\lambda_1 = \ \lambda_2 = 10, L_{idt} = L_{ctx} = 1$$

And in the CBAM model we have $\lambda_2 = 15$.

Denoting $(x, a) \in A\ X\ I_A$, $(y, b) \in B\ X\ I_B$, $G_A(x, a) = (y', b')$ and $G_B(y, b) = (x', a')$:

$$L_{LS-GA} \ = (D_A(x, a) - 1)^2 + D_A\big(G_B(y, b)\big)^2 + \ (D_B(y, b) - 1)^2 + D_B\big(G_A(x, a)\big)^2$$

$$L_{ctx} = \left|\left|w(a, b') \odot (x - y')\right|\right|_1 + \left|\left|w(b, a') \odot (y - x')\right|\right|_1$$

When $w(a, b')$ is the elementwise minimum (i.e., the intersected masked part) and $\odot$ is elementwise multiplication.

We trained InstaGAN for 300 epochs, using standard data augmentations (random cropping, flipping etc.).

The model we present is the baseline model, and we'll note that a possible reason for the collapse of the CBAM model (as shown in figure 3) is the high weighting of the cyclic loss which takes over as the dominant factor of the whole loss, however other hyperparameters settings we've tried has shown worse results.

Figure 4: some example results of the final floods model (no CBAM).

3. Stormy weather - In recent years, our world has witnessed a profound shift in weather patterns. While we've often associated extreme weather with certain regions, the lines between these climatic boundaries are blurring. Stormy weather, once a distant concern for some, is now impacting areas where it was historically rare. From winter thunderstorms in Israel to intense hurricanes in atypical locations, our planet's climate is undergoing a dramatic transformation. We aim to show visually how an extreme stormy weather could have looked like even in some areas that haven't experienced one yet. To achieve that, we set domain A to be images of sunny locations, and domain B images of storms, rains, clouds and etc. We've used the style transfer method for the task. We've used the default least-squares loss and a pretrained VGG-19 model. We have also examined 2 regularization factors: Total Variation (TV) denoiser [6], and sharpness loss [13] defined as the mean of L_1 norm differences between x the input and:

$$blurred = avgpool(x)$$

Style loss uses gram matrix calculation within the loss. We added calculations for style and content losses in the convolutional layers and to make the results more real and less art-wise, we added content layers in conv layers that capture features relevant to realism.

VGG-19 model uses ReLu as an activation function, and contains max pooling layers, conv layers, batch normalization layers and fully connected layers. We added image normalization as a first layer with ImageNet mean and standard deviation which is the convention for image normalization in pre-trained models.

We ran the model for 500 epochs.

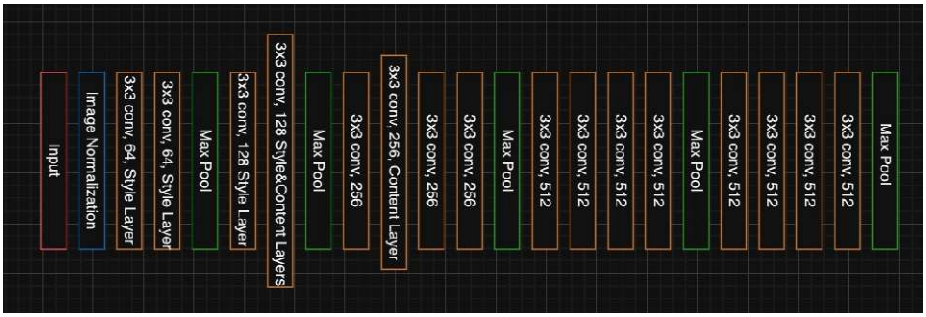

Figure 5: our style transfer model architecture.

We have found that the best combination is the following:

$$L_{storms} = \lambda_1 L_{style} + \lambda_2 L_{content} + \lambda_3 L_{sharpness}$$

When we set:

$$\lambda_1 = 1.8 * 10^5, \lambda_2 = 1, \lambda_3 = 0.1$$

Future work

In this work we suggest ways and training methodologies to take advantage of unpaired I2I translation technologies to raise awareness of climate change. Obviously, given time and resources, there are many more climate change effects that could be simulated in the same way as above (air pollution, forest fires etc.). However, each damage demands special care of gathering and preprocessing data (as far as we know, one could rarely find a prepared dataset), choosing the right model for best results and training it. In addition, in the CycleGAN-based methods, to create the "perfect personalization", when the damage could be integrated into every input scene, **one must create a highly diverse and large dataset to capture more types of input images**. Our models are very good for images that are similar in structure to the ones it trained on (e.g., the floods one is good for images that are similar those in CitySpaces dataset (but not necessarily from it!) but can sometimes struggle more on different images (especially on ones that noticeably differ from the dataset distributions).

In addition, due to time and resource constraints, we could not examine thoroughly as we would have wanted to the effect of adding CBAM to the generator. We did see in our experiments a possibility of faster and less noisy training by adding CBAM the way we did, but we did not have the time to resolve the collapsing in higher epochs.

References

[1] J. P.-A. M. M. B. X. D. W.-F. S. O. A. C. Y. B. Ian J. Goodfellow, "Generative Adversarial Networks".

[2] T. P. P. I. A. A. E. Jun-Yan Zhu, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks".

[3] M. C. J. S. Sangwoo Mo, "InstaGAN: Instance-aware Image-to-Image Translation".

[4] A. S. E. M. B. Leon A. Gatys, "A Neural Algorithm of Artistic Style".

[5] S. C. L. B. Martin Arjovsky, "Wasserstein Generative Adversarial Networks".

[6] C. R. V. a. M. E. Oman:, "Iterative Methods for Total Variation Denoising".

[7] M. O. S. R. T. R. M. E. R. B. U. F. S. R. B. S. Marius Cordts, "The Cityscapes Dataset for Semantic Urban Scene Understanding".

[8] B. a. Z. H. a. P. X. a. X. T. a. F. S. a. B. A. a. T. A. Zhou, "Semantic understanding of scenes through the ade20k dataset," *International Journal on Computer Vision.*

[9] R. K. a. R. P. a. V. Samadi, "FloodIMG: Flood Image DataBase System," *Kaggle.*

[10] I. G. W. Z. V. C. A. R. X. C. Tim Salimans, "Improved Techniques for Training GANs".

[11] J. P. J.-Y. L. I. S. K. Sanghyun Woo, "CBAM: Convolutional Block Attention Module".

[12] S. M. H. a. W. Z. a. W. X. a. W. S. a. G. E. Erfani, "ATLANTIS: A benchmark for semantic segmentation of waterbody images".

[13] A. K. H. M. B. N. Pierre Foret, "Sharpness-Aware Minimization for Efficiently Improving Generalization".