Deep Learning Workshop

Project Report - ShoreRestore

Yuval Alter, Yuval Lavi, Yonatan Ariel Slutzky

Problem Statement

Beaches are one of the most popular destinations for tourists and locals alike. Unfortunately, they are also one of the most heavily impacted ecosystems, with a significant amount of waste and pollution causing damage to both the natural environment and local communities.

Our project aims to build a web application where a user can upload a picture of a dirty beach scenery, and given the user's choice of cleaning intensity, the application will output an AI-generated image which is a re-imagination of the user's image as a clean beach scenery. By allowing users to visualize the impact of their actions on beaches, our app can encourage them to make more environmentally conscious decisions. The ability to show a cleaner beach environment through our app is also an interesting way to showcase the impact of beach cleanups and other conservation efforts.

An important note about our problem is that it is inherently hard – the definitions of a "clean" beach and a "dirty" one can vary greatly between different people and opinions, with some beaches being considered "clean" by some measures and "dirty" by others. Thus, constructing a computer model to differentiate between the two and transfer an image from one domain to the other is a tough task (even for a human).

Approach and Existing Frameworks

In our quest to refine beach images from cluttered to pristine, we harnessed the capabilities of advanced generative models. Our strategy was centered around unsupervised image-to-image translation, aiming to seamlessly convert images from the "dirty" domain, riddled with litter, to the "clean" domain, portraying a litter-free beach scene. Two preeminent models, CycleGAN and MUNIT, were enlisted for this task, each distinguished by unique architectural intricacies.

CycleGAN, conceived by Jun-Yan Zhu et al. in 2017, comprises of two generators and two discriminators, and designed to facilitate bidirectional image translation between the dirty and clean beach domains. The generators are responsible for transforming images, while the discriminators evaluate the authenticity of the generated images. Crucially, CycleGAN introduces adversarial loss to ensure the realism of the generated images and cycle-consistency loss to maintain semantic coherence during the translation process. Specifically in our task, the generator consisted of 9 residual blocks and its last convolution layers had 64 filters. The discriminator is a 70x70 PatchGAN which tries to classify if each 70x70 patch in their image is real or fake. This discriminator is run convolutionally across the image, averaging all responses to provide its ultimate output.

Multimodal UNsupervised Image-to-image Translation (MUNIT), introduced by Xun Huang et al. in 2018, features a more intricate architecture to accommodate multimodal translations. The core of MUNIT consists of an encoder-decoder network and domain-specific generators. The encoder-decoder network plays a pivotal role in disentangling content and style representations, allowing the model to discern shared content from domain-specific styles. This capability enables MUNIT to produce diverse outputs from a single input. Adversarial loss ensures the realism of the generated images, while domain-specific losses enforce adherence to the specified style in the target domain. In our specific task, MUNIT's encoder-decoder consisted of 6 residual blocks, with 64 filters in its bottom layer. The discriminator included 64 filters in its bottom layer and was 4 layers deep.

Datasets

To train CycleGAN – our model of choice, we employed data from the following sources:

Source 1 – 631 images of clean beaches

Source 2 – 76 images of polluted beaches and 76 images of clean beaches

Source 3 – 1000 images of clean beaches

Source 4 - 1287 images of clean beaches captured from a drone

Source 5 - 3500 images of polluted beaches

Source 6 - landscape images which include approximately 500 beaches

A key part of our project revolved around refining our training data. After collecting all of it, we manually filtered all images and classified each one as either “clean”, “dirty” or “unused”. We decided on a classification strategy that favored images who carry a lot of “signal”, i.e. were easier for a human to classify as either a “clean” beach or a “dirty” one. Specifically, we labeled as “clean” images that depict mostly sandy beaches, that were preferably empty, and included some water scenery. We labeled as “dirty” images that depict sandy beaches with a somewhat high level of pollution and/or garbage, that preferably also include some water scenery as-well as sandy scenery. We made our best to make sure the difference between our classes relied mostly on the pollution existing in the “dirty” class and not existing in the “clean” class. Finally, we labeled as “unused” images that either didn’t depict a beach at-all, were of bad orientation (e.g., shot from high altitude or from close focus), or depicted images at night hours (we decided to focus on daylight images). We also labeled as “unused” images that were ambiguous – i.e., images we couldn’t label as “clean” nor “dirty” with high enough assurance.

After the first round of model training, we enriched our training data by adding images found using a Google images web scrapper, with the goal of adding more images of dirty beaches. The python notebook used for scrapping is available in the project’s GitHub page. After the second round of model training, we further refined our training data by filtering the present images in a more aggressive manner, in hopes that the filtered images will better capture the essence of “clean” vs. “dirty” beaches.

After preprocessing all the data as detailed above, we enriched our dataset by adding all the horizontally flipped copies of the original images. Next, each image was rescaled to 512x512 pixels and then a random square of 512 pixels was selected from it to be used in training. These procedures made our training dataset more heterogenic, helping our model to avoid overfitting and be more robust.

Training and Selected Results

In total we trained three different CycleGAN models, using the three versions of our dataset - original, google-images enhanced and aggressively filtered. We also experimented with one MUNIT model. The training process took place in Google Colab, using premium resources (V100 GPU). Training each of the models for 200 epochs took approximately 5 days to complete. Due to the lengthy training process and the limited number of resources available to us, we avoided experimenting with other architectures for the generators and discriminators, and instead focused on the architectures used in the original papers for CycleGAN and MUNIT.

We continue to present some selected results generated from the CycleGAN models (the MUNIT model did not reach any meaningful results). The following are some successful cleaning results obtained from the second model:

Real

Fake

We assess that the model was especially successful in cleaning waste that had unique and noticeable colors, i.e., waste that stands out when viewing a beach landscape. The model was less successful when the image was from an angle that makes the trash capture a large part of the image. The following are some unsuccessful cleaning results obtained from the second model:

Real

Fake

Real

Fake

The first and third models were trained on an imbalanced dataset, where the clean images were taken with a frontal view of the sea line, while the dirty images were much more focused on the sand. This resulted in a tendency to add more water to images when “cleaning” them and moving the shoreline. For instance,

When trying to convert clean images into dirty images, none of the models were able to produce satisfying results. Images which were "littered" by our models usually became darker and sometimes saw a colorful noise added to them. We interpret this as the model learning that waste on a beach comes in the form of uncharacteristic color on the sand, but failing to create objects that can be understood. For instance,

Application

To showcase the capabilities of the second model which had the best results, we created a user-friendly web application using Streamlit. The application provides an easy-to-use interface that lets users translate images of dirty beaches into their respective reimaginations as images of clean beaches. A phenomenon we encountered when experimenting with our model is that reapplying it over the output produces significantly better results for some images (specifically ones that weren't cleaned successfully after the first run). Therefore, we decided to allow users to choose the desired cleaning intensity out of 3 possible intensity levels. The difference between the different intensity levels is the number of rounds the input goes through the model; Light intensity, which is the standard mode, applies the model to the input for a single run. The higher levels reapply the model over the output, with the moderate intensity running for a total of 3 runs, and the extensive intensity running for a total of 5 runs. The following is a showcase of the different levels, starting with the input on the left, and going up to the extensive intensity on the right:

Github Structure

Our work is organized into 2 separate GitHub repositories. We decided to split the repositories to make things easier to navigate. The first repository serves as the main one, and is comprised of the files used to build our models, as-well as the documents describing our work. The second repository serves as our dedicated application repository and is comprised of the files used to build our Streamlit application.

Main Takeaways and Future Works

We believe our work can be improved further by focusing on three main areas, addressing some points we learned along the way. First, one can choose to tackle the training data and enhance it. Our work relied on data that was collected online, with many of the images not fitting the “ideal” representations of “clean” / ”dirty” beaches, disturbing the model’s performance. In an optimal scenario, the data collection can be less forgiving in terms of what filtering is employed to determine the images to be used and can require all used images to reach a stricter standard. This of course will require many more images to begin with and more time on data refinement and classification. Second, one can choose to treat the approach we employed to solve the problem. As we came to realize, beach litter can sometimes dominate images of dirty beaches, to the point where most of the image is comprised of litter. Thus, an image translation solution can sometimes miss the target, which in the case above is to completely remove all the litter, instead of changing the style of the image. A wiser solution can be first to employ an image detection solution which will identify the elements to be removed, and then employ an image translation solution (after the unwanted elements have been removed). This new approach is semi-supervised and will require the training data to be of a different nature, since detecting the litter requires the data to be labeled (e.g., which pixels contain litter). Finally, one can also choose to rethink the modelling solution itself. The number of resources available to us was limited and thus we didn’t try to scale our models, finetune them further or try other models. Extending the size of the trained models can lead to an improvement in results, with the obvious caveat of more required resources.

In addition to the above, one can also choose to extend the work to fit other types of ecosystems such as forests, public parks, etc. This extension will further emphasize the impact humanity has on the environment and what is the true potential of keeping it clean and avoiding pollution.