**Computer Graphics**

**Mid Evaluation of Final Project**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks | | | | |
| **Name** | **Student ID** | **WeChat ID** | **Email** | **Contact Number** |
| **Wang Hao** | **515021910060** | **wanghaochengdu** | **wrystal@sjtu.edu.cn** | **15821160972** |
| **Wang Xiao** | **515021910141** | **Alanxiao0412** | **1042802231@qq.com** | **15821803867** |
| **Qiang Zhiwen** | **515030910367** | **Qzw688688** | **qlightman@163.com** | **17775255690** |

**Project Name/Title**Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Our project tries to implement an unpaired image transformation using adversarial networks with constraints, the cycle-consistency. During this process, the generator network competes with the discriminator network. After the training, the desired generator can achieve successful image transformation.

**Project Abstract**Currently, most of the image transformation is about one-to-one mapping, which in other words, pairwise images are used to train the model. In in this project, our goal is to implement an unpaired training. This training is to learn a mapping *G: X -> Y* (*X, Y* are two groups of images respectively, and *X* is the source domain, *Y* is the target domain) such that in ideal case, the generated image *G(X)* has the same distribution of the images in *Y*. And in this way, the generated image has some features of the images in *Y*.

The basic idea of implementation is to use adversarial networks. The generator network tries to produce images from the desired distribution, and the discriminator network needs to distinguish whether an image is a real image from the target domain or a fake image generated from the source domain. And in this way they compete with each other, finally we will get a desired generator. Since there is large capacity, a network can map the same set of input images to any random permutation of images in the target domain, any of the learned mappings are able to satisfy such distribution, which in this way we cannot always get the desired output images. As a result, more constraint needs to be added. The cycle-consistency, if we transform from source distribution to target and then back again to source distribution, we should get samples from our source distribution. And with the help of this, successful transformation can be achieved.

**Current Initial Results**

Till now, we have set up the GPU-Version-tensorflow both on Windows and Linux platforms. Also, we tried to pick some images from the apple2orange dataset to train a sample generator. But since our PC’s performance is not good enough and the training process normally takes plenty of time (several days maybe), we can only train a very small data set in acceptable time now. Although it can generate a *G(X)* from *X*, it does not look very real. Currently, we are trying to train a complete generator with the help of a high performance server and see what we can get.

So far we have achieved:

1. Read and fully understand the paper
2. Set up the GPU-Version-tensorflow both on Windows and Linux platforms
3. Pick some images from the apple2orange dataset to train a sample generator.
4. Obtain a rough result from training a very small data set.

**Expected Results Description**

Our project’s overall goal is to capture special characteristics of one image collection and figuring out how these characteristics could be translated into the other image collection, all in the absence of any paired training examples. And to do this, we try to use the CycleGAN method which was proposed in the paper. However, the proposed method is quite primitive, there can be many efforts to improve the performance such as adding some ‘skips’ into the training, etc. Our final result should be several generators which can translate some specific types of images into another type such as horse->zebra.

Below are roughly all the goals we need to achieve in our project(some mentioned above have already been done).

1. Read and fully understand the paper
2. Set up the tensorflow working environment
3. Select a small dataset to train a sample generator to test the algorithm
4. Use a server to train a complete generator to test the performance of the proposed method
5. Go through the source code and try to make some improvements to faster the training process.

**Reference Paper**

Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks, Jun-Yan Zhu ∗ Taesung Park ∗ Phillip Isola Alexei A. Efros, arXiv:1703.10593v2 [cs.CV], 5 Oct 2017, 20.