CS 464
Introduction to Machine Learning
Progress Report

Görkem Yılmaz - 21601927
Arda Türkoğlu - 21601187
Ege Turan - 21502441
Çağrı Orhan - 21503102
I. Umay Durur - 21502041

16 November 2020

1 Introduction & Background Information

"Translating" is the most common problem in computer vision and computer graphics, which means learning the mapping between an input image and output image [1]. Usage areas of image-to-image translation extend to many places. Some of these are style transfer, object transfiguration, season transfer and photo enhancement. So, we set out to do a project in this area as a group of five by using Generative Adversarial Network (GANs). Generative modeling is an unsupervised learning task in machine learning to generate new output by learning and detecting the patterns in the input [2]. Thanks to GAN, we organize our model to learn like supervised learning by generating two sub-classes which are the generator and discriminator as in Figure 1. Generator generates the output as its name implies and discriminator tries to classify if the output image is real or not by learning a loss so that the modal can minimize this loss.

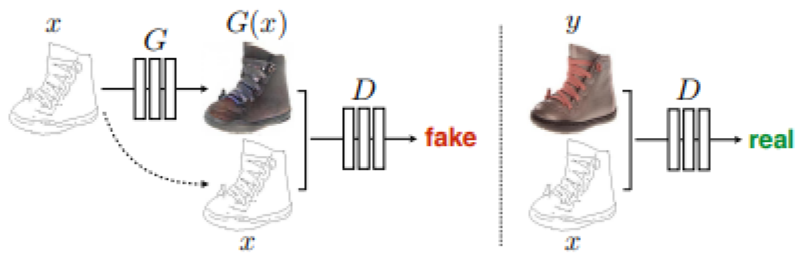

Figure 1: GAN Model Demonstration

For our model, we also go one step further and use Conditional Generative Adversarial Network (cGANs) to learn a conditional generative modal which is perfectly fit for image-to-image translation [3]. As shown in Figure 2, the main purpose of cGANs is to use labels to improve performance of GANs because in GANs, there is no control over data to be generated. In this way, label y is added to the generator as a new parameter to get higher performance when generating outputs [4]. This is why we are using cGANs in our model. The cost function of cGAN is exactly same with GAN.

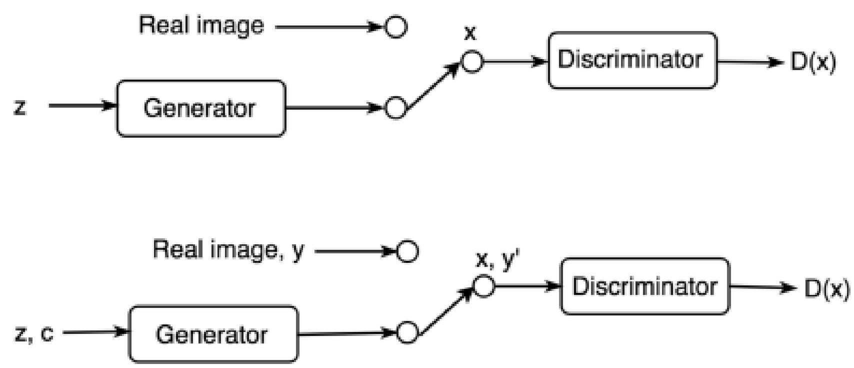

Figure 2: cGAN Elaboration

In image-to-image translation, each pixel of input is considered as a separate and independent pixel which called per-pixel classification so that the output of the generator is considered as unstructured because again each pixel is independent from each other. The good thing is cGAN learns structured loss so we can penalize any structure that cGAN found to get an output that is indistinguishable by the discriminator. As a result, we will use generative model in our project to learn the mapping from input to output image. We will use day to night and CitySpace as our dataset. To make it better cGans will be used to learn structured loss from label maps to synthesis photos by hoping to beat the discriminator.

2 What Have We Done So Far

Once we have agreed upon working on GANs (General Adversarial Networks) we started to do a detailed research about pix2pix and cyclegan methods. After we reviewed our research as a group, we decided to focus more on pix2pix model and we planned our work based on pix2pix model. Once it is decided, we tried to fit the dataset we wanted to use in the model to pix2pix format. At this point, we started to encounter some problems. One of that problem was that we had trouble with the size and the format of the dataset. This was a problem mainly because pix2pix model requires an image that is combined from two images to train. One of those images is the image that a segmentation method is applied and the other one is the ground truth image. At that point of the project, we were working with the cityscapes dataset and our main goal was to get a pix2pix model output from the segmented image input. As mentioned the problem was the necessity of an image that is combined from a ground truth image and a segmented image. In order to make the model work with the cityscapes dataset, we implemented a code script that divides our dataset as a bunch of image couples that is in the right format for pix2pix model. The implementation of our script took some time but when it was worked without a problem, we created ourselves a template dataset that we can use in pix2pix model training. Also, with the usage of our script, sizes of the images in the dataset are set to the required image size for pix2pix model. At this point, our sample dataset was ready and we were ready to train the pix2pix model.

Training procedure of the pix2pix model takes a lot of time and in order to efficiently use our time and computing power, we purchased Google Colab premium and Google Drive storage and we started our training procedure. Since the training is complex and takes plenty of time, we used a checkpoint system in the training part. It was useful because in any case if the training procedure is interrupted of any other kind of error happened that would destroy the training, we would not lose any training progress. So basically, we could start the training from the epoch where our program left the training. By using this logic, we trained the pix2pix model using the cityscapes dataset with 170 epochs. Each iteration in each epoch gives loss of G-Gan,G-L1,D-real,D-fake. The images that are generated during the training process can be seen in Figures 3, 4, 5 and 6.

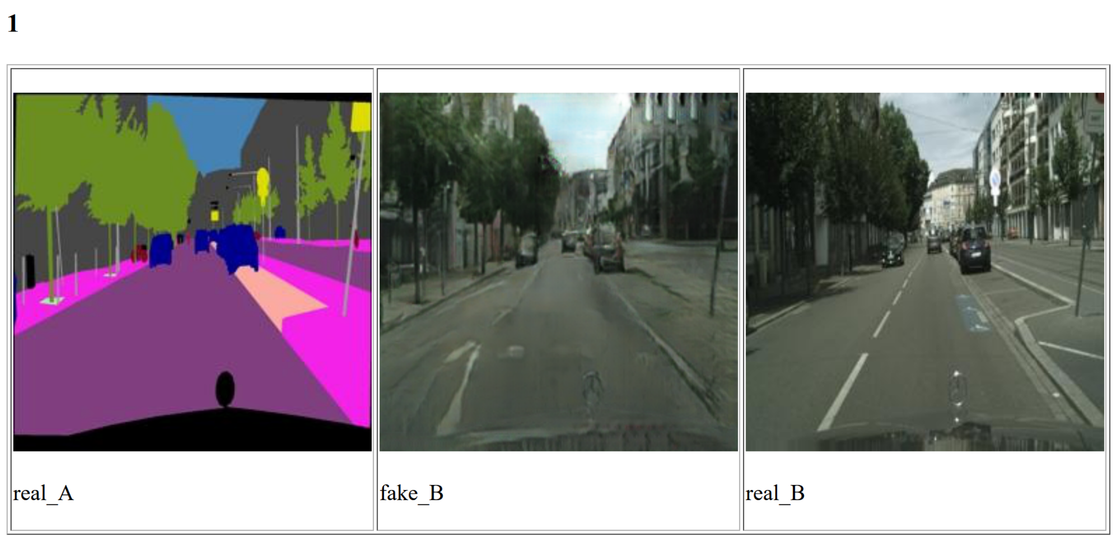

Figure 3: The First Epoch

We implemented a script which takes the loss of each iteration and plot them as loss vs epoch. Also, we did several contrasting tests and we saw the epoch versus loss information. Eventually, we compared the generated results with other trained models with various epoch sizes. Finally, we implemented another script that takes the accuracy and loss information, hence, executes the plots of both accuracy and loss, which can be examined in Figure 7, and we took the outputs.

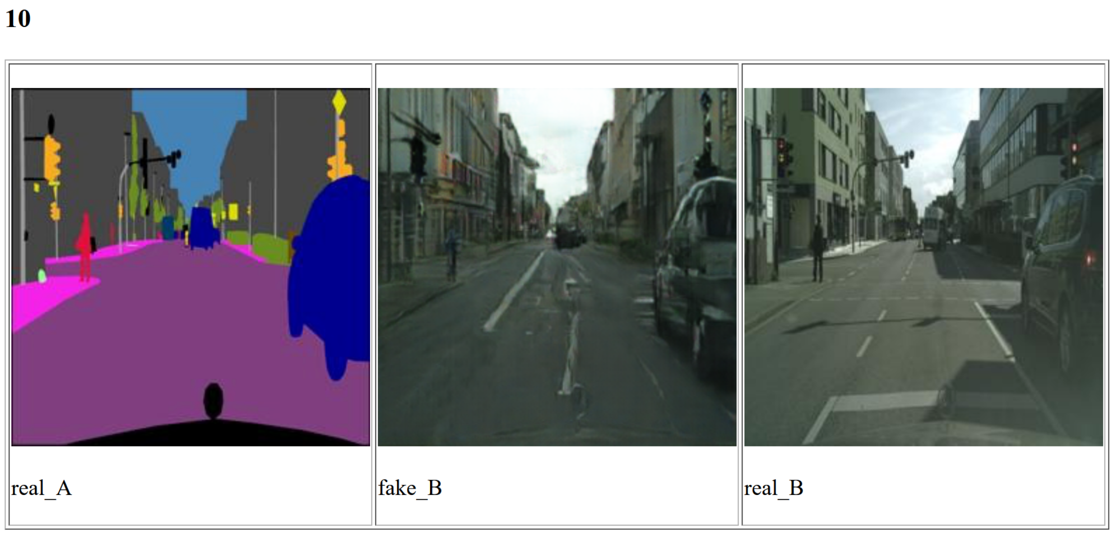

Figure 4: Epoch 10

Figure 5: Epoch 110

143

Figure 6: Epoch 143

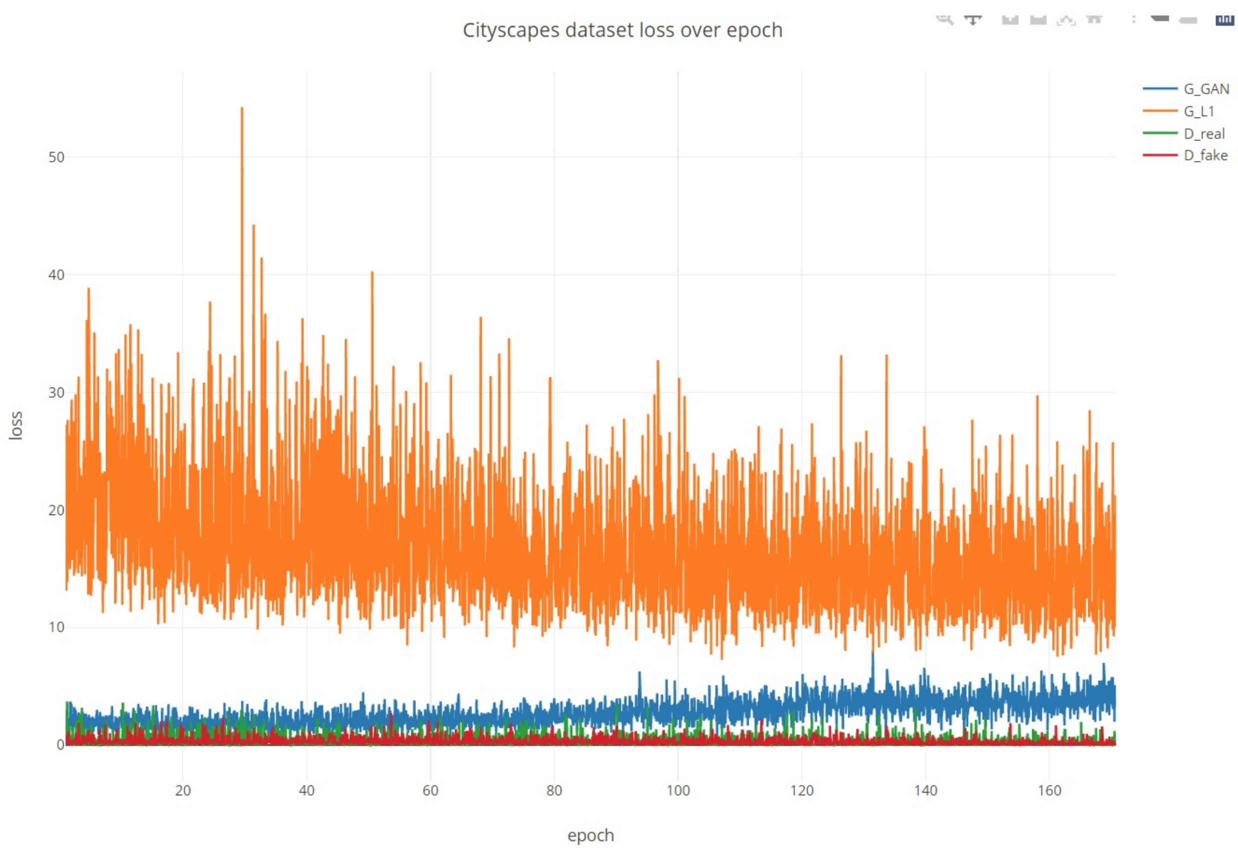

Figure 7: Loss vs Epoch Plot

3 What Remains to Be Done

As we finished all the things that mentioned in the previous part, our next step is going to be to train the model with another dataset that was not acknowledged in the paper of pix2pix. At this point, our top preference is the dataset that contains day and night images. Our goal by training the pix2pix model with this dataset is to generate night images from the day images or the other way around. If we will generate night images, then night images in the dataset will be the ground truth images. As we formatted the previous datasets that we used, we will again use our script to set the size of images in the dataset and we will again combine the ground truth and segmented image pairs. We will convert our dataset in to a set of those pairs of images. After the training is over, we will do testing with various cases as inputs and we will compare the generated output results (accuracy and loss) with other datasets and models. Also, in the final report of the project, we will compare the accuracy and loss results with different epoch sizes. We will report our accuracy. Finally, by using this GAN model and a landscape dataset, we will try to convert summer images into winter images and winter images into summer images. Again, we will compare the accuracy and loss results with other results that we recorded.

4 Description of the Division of Work Among Teammates

In our project, each teammate and their responsibility are as follows:

Ege Turan & Çağrı Orhan: They are responsible for the procedure of training the pix2pix model with desired datasets and other inputs.

Arda Türkoğlu: His responsibility is to report the accuracy and loss after each training procedure.

Umay Durur: He is responsible to develop and use the script that resizes the images in the dataset according to the required format and other works to prepare a dataset for training the pix2pix model.

Görkem Yılmaz: He is responsible to compare the results that we will collect with other results and models.

References

[1] Y. Hao, "Image-to-image translation," March 18, 2019.

[2] J. Brownlee, "A gentle introduction to generative adversarial networks (gans)," July 19, 2019.

[3] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial nets," 2017.

[4] J. Brownlee, "Gan — cgan infogan (using labels to improve gan)," June 3, 2018.

[5] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "pix2pix," 2017.