New Dogs Images with GAN

Group 1: Chuan Su, Diego Guillen-Rosaperez
2020, 12th January

Introduction

With the current access to high computational resources and meaningful image datasets, in addition to the advancements in deep learning methods, it is possible to develop algorithms that emulate creativity on a computer. Tasks related to image classification are relatively simple with the correct dataset. But, what about creating a new one with unseen images.

One technique to achieve it is by using a Generative Adversarial Network (GAN). It is composed by two neural networks: a generator that tries to generate data that looks similar to the training data, and a discriminator that tries to tell real data from fake data. During training, both of them compete against each other. An analogy is to compare the generator as a criminal trying to make counterfeit money realistic, and the discriminator as a police investigator trying to distinguish real money from fake [1].

On one side the generator receives as input a random distribution, such as Gaussian, and outputs a tentative image (in this case). Then this fake image goes to the discriminator as an input, so it can guess if it is fake or real. During training, the goals of both neural networks are opposed. The generator tries to produce images similar enough to trick the discriminator, while the discriminator tries to distinguish correctly fake images from real.

Therefore, this technique could be used to extend our imagination, and develop new pictures that may look like dogs.

Problem description

Dogs are a very interesting partners to see and people like to take pictures of them. Nevertheless, our knowledge about the potential pictures of dogs is limited, which makes us like to go further than human taken images. Therefore, by using GANs, we could go beyond our human creativity, and see a unique set of images centered around our hairy partner.

Methodology

The dataset we used in our project was from Stanford Dogs Dataset and training was conducted on Kaggle where 30 hours GPU was offered weely.

We started our project by data preparation and image preprocessing, which includes cropping and resizing images according to the bounding box specification as well as image data normalization.

Then, an initial GAN algorithm was developed. Our model was inspired by DCGAN (deep convolutional GANs) guidelines for building stable convolutional GANs in which it suggests to use Batch Normalization in both the generator and discriminator. However we discovered that adding Batch Normalization or Dropout layer in discriminator generated very unstable results. We also experimented SELU activation in generator model layers and confirmed with DCGAN guidelines that ReLU tends to produce more stable results. During training we experimented with adam and rmsprop optimizer and both tended to provide similar results.

We experimented first training the algorithm with grayscale images, with which we could obtain dog like shapes. Nevertheless, the details the level of details were hindered and most of them shown only the contourns. A next iteration was to train the algorithm with only one dog breed still in black and white. The details improved, but the images were mostly blurry, since the dataset was significantly reduced.

Later, the model was trained with the complete dataset (all breeds) and using all their RGB layers. The results of this experiment are shown below.

Results

We trained our GAN model with 400 epochs. The training time was around 5.5h on a NVidia K80 GPU.

The results in the first 200 epochs looks promising while the results produces between 200 and 400 epochs are very unstable. The images produced by GAN started becoming less diverse and in the last 100 epochs they tend to become identical! And our GAN was suffering from model-collapse problem.

We tried to adjust our model by adding Batch Normalization and Dropout layers and tune hyper-parameters such as learning rate and even with different seed/noise but it was not getting improved.

The screenshots below presents the results of our GAN model at epoch 181 and epoch 398. As you can see that at epoch 398 our GAN model produced nearly identical 25 images.

Image 1: Generator and Discriminator Loss, Epochs 0-186.

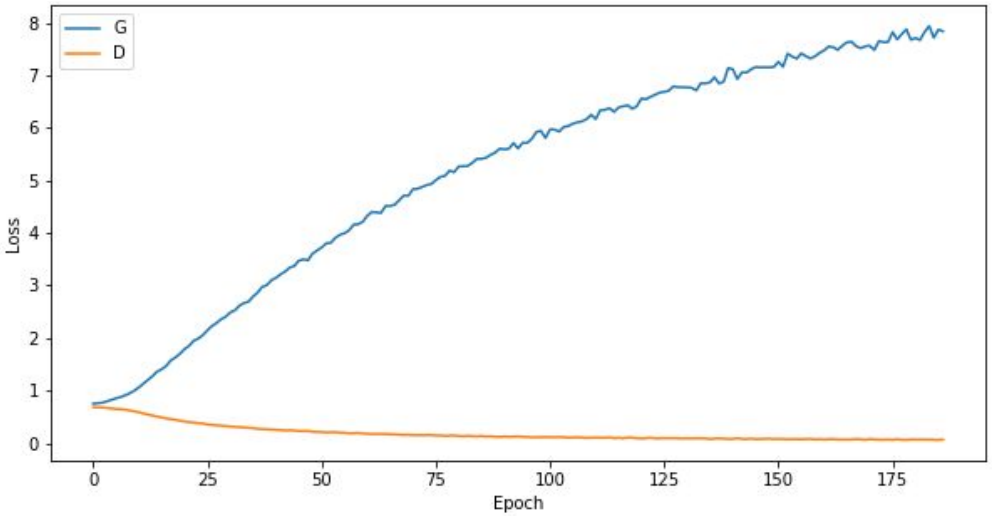

Image 2: Generated images after 186 Epochs.

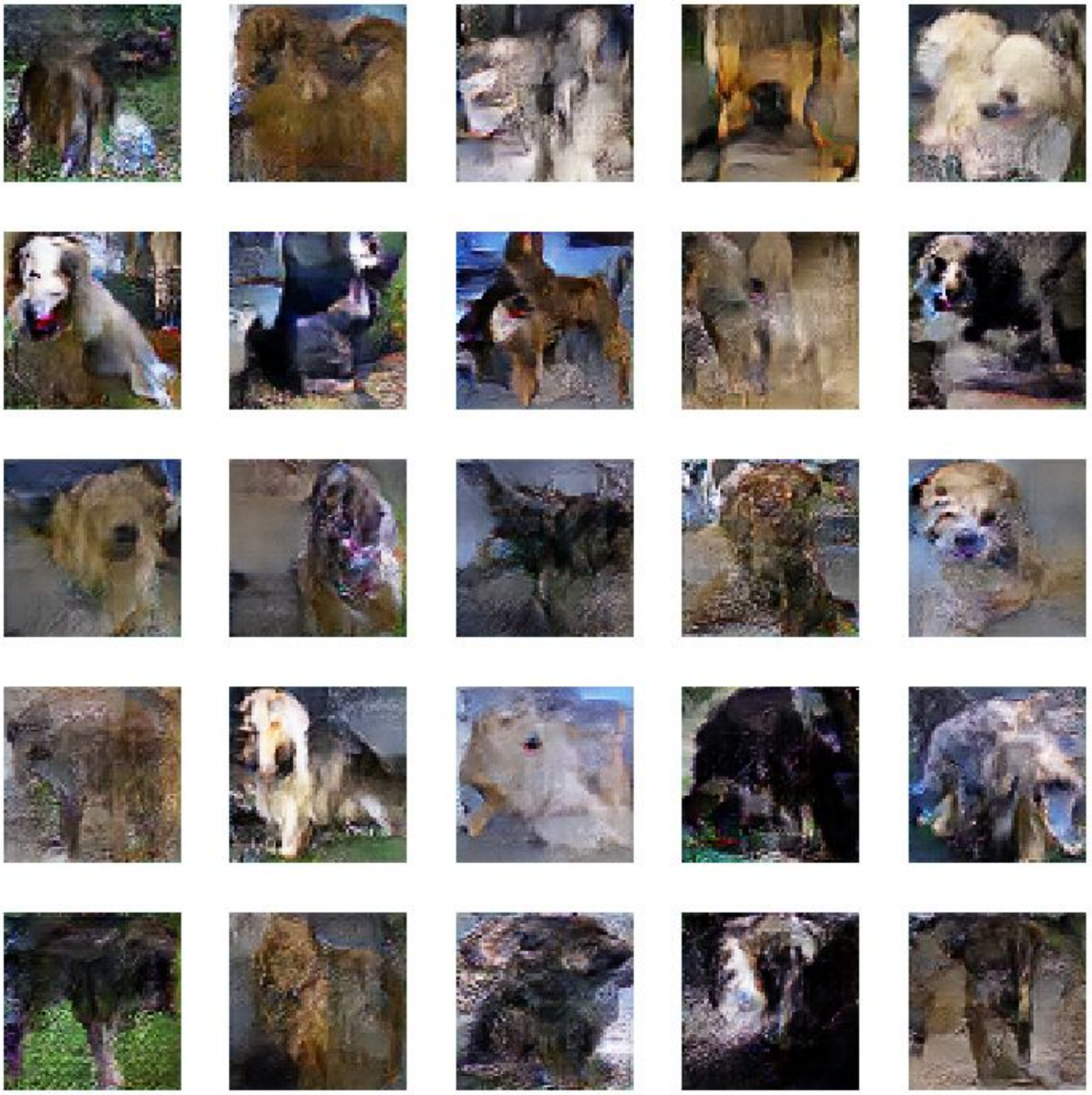

Image 3: Generator and Discriminator Loss, Epochs 0-398.

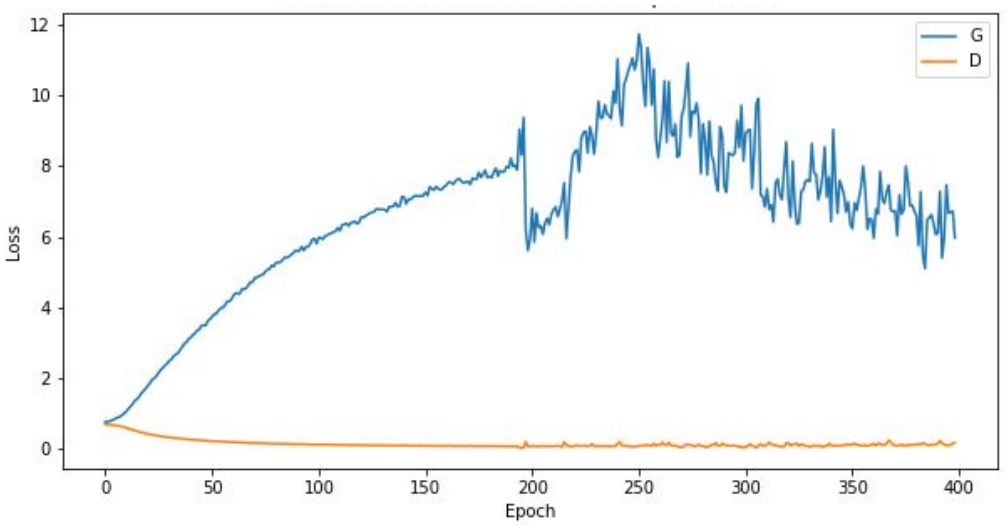

Image 4: Generated images after 398 Epochs.

Conclusion

This project implemented successfully DCGAN to generate dog images with stanford dogs dataset. Moreover we followed most of the DCGAN guidelines, except we removed Batch Normalization layers from discriminator. It was additionally founded an epoch range when the model-collapsed.

Bibliography

1. Aurlien Gron. 2017. Hands-On Machine Learning with Scikit-Learn and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems (2nd. ed.). O'Reilly Media, Inc.