Udacity Machine Learning Engineer Nanodegree

Capstone Project Proposal

Svetoslav Paregov
March 21, 2019

Single Image Super Resolution through a CNN

Domain Background

Single image super-resolution is the task of inferring a high-resolution image from a single low-resolution input. Image resolution enhancement is something that we used to see in many movies, but it is not something trivial to achieve. There are a lot of algorithms used in different software products that increase the image resolution with different degree of success.
Machine Learning has been used in many different image processing tasks with Convolutional Neural Networks. Detection of different objects, face recognition and other. In the last few years it is also used to improve the image resolution.

What it can be used for – increasing the size of pictures from old phones for printing or some other purpose; Increase the quality of old security footages; Use it as base for other Machine Learning tasks to provide better quality images;

There are a lot of researches on the topic from different universities. I'm basing my project on one from Max Planck Institute for Intelligent Systems, Germany [1].

Problem Statement

Traditionally, the performance of algorithms for this task is measured using pixel-wise reconstruction measures such as peak signal-to-noise ratio (PSNR) which have been shown to correlate poorly with the human perception of image quality. As a result, algorithms minimizing these metrics tend to produce over-smoothed images that lack high-frequency textures and do not look natural despite yielding high PSNR values. In other words, for the computer looks good but not for the humans. A model or algorithm that is improving for human perception is needed.

I propose a CNN model that will be optimized for sharper and close as possible to the reality images. I hope to achieve this with correctly implemented loss functions. Still this is a hard problem since during the scale down a lot of information is lost and there so many variations when scaling up. So, part of the missing information should be synthesized by the CNN.

Datasets and Inputs

I have the MSCOCO and CelebA datasets which will be used separately to train the model.

MSCOCO dataset has images from few years and combined they are more than 250,000. This dataset contains all kind of images – animals, people, cars, buildings and many more.
CelebA has a little over 200,000 images and contains images of people.

The low-resolution image that I'm going to feed into the model input is 32x32 pixels and the output image will be 128x128 pixels. Selected resolution increase ratio is 4 times.

Solution Statement

I propose to use a Convolutional Neural Network (CNN) that learns how to synthesize the missing pixels. I'm going to use Generative Adversarial Networks (GAN) to train my model. Training on different datasets with different loss functions will be performed and tested.

Benchmark Model

I can compare the results with different software products that increase the image resolution. There are also few already trained Convolutional Neural Networks for the same task. I plan to train the model using different loss functions and compare the results.

Evaluation Metrics

Evaluation can be done visually or by feeding the enhanced images to an object recognition model.

Project Design

First thing is to split the dataset. Selected ratio is 75% for training set, 20% for validation set and 5% for test set. I selected such percentage for the sets since the model need a big training set and doesn't need so much for the testing set since I'm going to do mostly a visual inspection. Even 5% on 100,000 images are still 5,000. I am not going to use all of them to manually check how the model performs. If possible, I'm going to do an automated test with a pretrained object recognition model.

I'm going to start with the layers proposed in the paper [1] and from there can change the model for improvements.

First approach to use is training the model directly on the training set for let's say 1000 iterations. This is the standard approach for training neural networks.

Second approach is using GAN. First, I'm going to generate some images with the generator and combine them with the real images from the dataset. Then train the discriminator. Once the discriminator is trained, I'm going to train the generator to produce better images so it can fool the discriminator. I'll try first with training for around 24 hours on a single GTX 1080 ti.

Image pre-processing:

I'm going to filter the images so they have at least 384 pixels on the short side. Then the images will be cropped to a rectangular image using the short side for size. Cropped image will

be resized to 256x256 pixels, so they don't lose too much details. Selected size for low-resolution image is 32x32 pixels. Since our selected increase ratio is 4, that means the high-resolution image will be 128x128 pixels.

It is important to have the correct loss function so the CNN is trained restoring the correct details. I plan to use different loss functions in the process and compare them. So far, I have in mind a perceptual loss and texture loss functions. For the perceptual one I'm going to use VGG16 and VGG19 and will see which one will give better results.

References

1. https://arxiv.org/abs/1612.07919