Final project topic

CV Super Resolution

EE 541: Spring 2024

Authors: Zeyu and Dr. Brandon Franzke

Introduction and Background

Welcome to Computer Vision (CV)! A super important field of deep learning that helps your computer to "learn", see and understand the contents and intricacies of images and videos. Even though an image or the frame of a video might *seem* like a very simple thing to look at, the complexities of it are boundless. At an abstract level, the objective or target of CV would be to observe images and infer something about it. It is a very broad area of study and through this project, we'll be spending a good amount of quality time in understanding images and solving real-world problems. One of the most recent hot topics of vision is single-image super resolution. In essence, the aim of such a model or system would be to enhance a low-resolution (LR) image to an image of higher-resolution (HR). Figure 1 is an illustration of such.

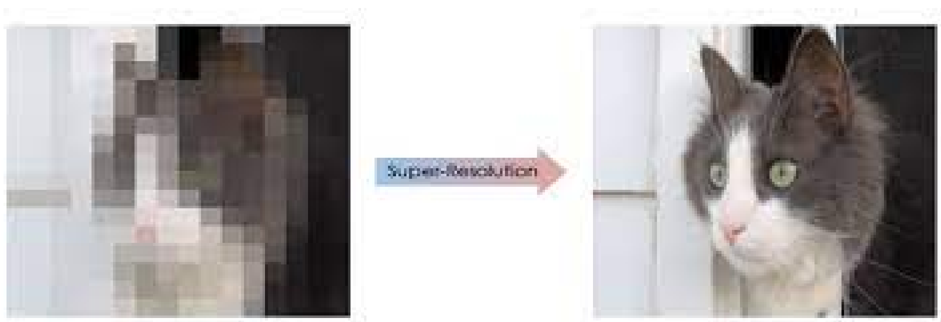

Figure 1: Super Resolution

Another classic area of vision research is image denoising. As the name suggests, it's the process of removing noise from an image. Most images are corrupted with some kind of noise of a particular distribution like white noise, Gaussian noise, salt and pepper noise, or some random noise, which degrade the quality. As the demand for pleasing images continue to increase, this area tends to get more traction. But, you're free to explore vision and it's sub-problems through this course project.

Dataset

We present to you the DIV2K dataset that hosts a variety of down-sampled RGB images and their corresponding high resolution images. The main idea of image down-sampling is to remove an equal number of rows and columns of an image (by a certain factor "x"). For example, a down-sampling factor of 4 would throw away every fourth row and column of the image. This reduces the size of the image and degrades the quality too, making it an LR image. But, the opposite i.e up-sampling, wouldn't exactly do the same that you would expect. Even though it increases the dimensions of the image, the image quality still remains poor (or maybe slightly better through certain interpolation methods).

DIV2K has the following structure: It is divided into 800 training, 100 validation, and 100 test images. All the diverse images have been randomly picked up from the internet paying special attention to the image quality.

Images ranges from people, handmade objects and environments, to flora and fauna, and natural scenes including underwater and dim light conditions. The HR images are in the "DIV2K/DIV2K_train_HR/" folder that houses the 800 train images. The corresponding validation and test images are in the "DIV2K/DIV2K_valid_HR/" and "DIV2K/DIV2K_test_HR/" folders, respectively. Each image is marked as "0001.png", "0002.png", etc. Another track of this dataset holds the down-sampled images, either by a cubic downgrading operator or randomly down-sampled. They are marked as "YYYYx2.png", or "YYYYx3.png", or "YYYYx4.png", where YYYY is the image ID and x2, x3, and x4 represent down-sampling by a factor of 2, 3, and 4 respectively.

Accessing the Data

In this, we'll be using the **HR and the bicubic down-sampled images** from the DIV2K dataset.

Windows users: First install "wget". Follow this link. to get it installed.

Linux or Mac users: wget is pre-installed on most Linux distributions and MacOS. To check if it is present, run "wget -v" in your terminal. If you get a positive output, wget is installed. Otherwise, it will print "wget command not found".

Once completed (for Windows/Linux/Mac users), run the commands (command prompt for Windows or terminal for Linux/Mac) to download the dataset to your desired directory.

```
wget http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_train_LR_bicubic_X2.zip
wget http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_valid_LR_bicubic_X2.zip
wget http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_test_LR_bicubic_X2.zip
wget http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_train_HR.zip
wget http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_valid_HR.zip
wget http://data.vision.ee.ethz.ch/cvl/DIV2K/DIV2K_test_HR.zip
```

This will download the train, valid, and test folders of the HR and bicubic DIVI2K dataset. Also, you're encouraged to experiment with different down-sampling rates. Replace X2 with X3 or X4 in the url to download the respective image folders. After that, unzip to get access to the dataset under consideration.

Example Project Objectives

Super Resolution: The DIV2K dataset primarily focuses on single-image super resolution. In this, as mentioned earlier, our target would be to obtain an HR image, given an LR image. So, from the dataset, an (input, output) pair would be an (LR, HR) imaging pair. Standard practices would be to make use of deep Convolutional Neural Networks (CNNs) to achieve this. You're also encouraged to look into an Encoder-Decoder network too. Here, the encoder is a CNN block that takes an input image and outputs a fixed vector representation, which acts as the input to the decoder that returns an image of the same size as that of the input image. But, this is purely unsupervised. Autoencoders (AEs) and Variational Autoencoders (VAEs) are also of good interest here. Generative Adversarial Networks (GANs) are the state-of-the-art when it comes to super resolution. You could also experiment with the classical ResNet's variants, since the skip connections will aid in preserving the local information while doing super resolution. But the caveat is that ResNet, as a model, is very heavy in terms of parameters. Appropriate GPUs are required to train your models. Image pre-processing including the normalisation is very important.

Image Denoising: CNNs have been a classic when it comes to noise removal. You should consider using the bicubic section of the DIV2K dataset that you've downloaded to design your deep learning system. Experimenting with the same set of networks mentioned earlier could be a good start. However, data augmentation is a very crucial step. You could add some more noise, crop, add contrast, increase/decrease brightness, etc. To evaluate your model performance in this case, Signal-to-Noise Ratio (SNR) or Peak SNR (PSNR) are good evaluation metrics to quantify the amount of noise present.

Tips

Ensure that sure you're using a GPU to train your models. Use appropriate data augmentation strategies, evaluation metrics, optimizer, regularizer, learning rates and scheduler, etc. Choosing a loss function is very important when it comes to solving a vision problem since it's important to note that you'll be dealing with pixels, context in the image, local and global information, etc.