Super Resolution of Images using Sparse Coding Techniques

Impulse 2021 : SPS Chapter, IEEE NITK

Description : In this problem, you will be given a Low Resolution Input Image. Your goal is to Upscale it to a Higher Resolution Image using Sparse Coding Techniques. The Problem is divided into 8 sub-parts. Follow the instructions and make timely submissions to stay on top! Happy Learning and Goodluck!

General Instructions :

1. *Answer each of the following questions and submit each part individually.*
2. *You are allowed to use any resources available online, but please avoid Plagiarism.*
3. *Hard Coding is preferred over Inbuilt Libraries. Make sure your code is legible.*
4. *You should consider using only the following libraries : Numpy, Scipy, Matplotlib, OpenCV*
5. *You can download the required files from the link below :*

Resources for the Competition

Submission Guidelines :

1. *Upload your College ID card (all 3 members) using this form.*
2. *Create a folder with name :* ***Impulse#team_number*** *on your Google Drive.*
   Submit the folder link using this form. Also, provide edit access to the following emails.
   saikumardande2001@gmail.com, adichand20@gmail.com, kaushikalwala1729@gmail.com
3. *Create 8 different folders for each task with the name* ***Task#task_number****.*
4. *Submit the tasks as soon as you finish. We will be keeping track of the time of submission.*
5. *For each task submit all the relevant files and outputs.*
6. *Add appropriate comments & describe each scripts function with a short description.*
7. *While writing the code, put the required input files in the same working level directory.*

These are the Tasks which need to be performed in this problem statement :

Task 1 :

Consider the system of equations $\mathbf{A}\boldsymbol{x} = \boldsymbol{y}$ where $\mathbf{A} \in \mathbb{R}^{m \times N}$ and $m < N$. Let $\boldsymbol{x} \in \mathbb{R}^N$ be a solution to the system. Now, suppose it is known a priori that the solution with $supp(\boldsymbol{x}) = \{j \in \{1, \cdots, N\} \mid \boldsymbol{x}_j \neq 0\}$ has cardinality s, where $s << N$, i.e., the vector $\boldsymbol{x}$ is sparse. Design a method to find such a solution $\boldsymbol{x}$ with lowest possible error given the output vector $\boldsymbol{y}$ and the Matrix $\mathbf{A}$.

Along with a well documented code, submit the output of your code for the following inputs :

1. $\boldsymbol{y} = \begin{bmatrix} 1.65 \\ -0.25 \end{bmatrix}$ and $\mathbf{A} = \begin{bmatrix} -0.707 & 0.8 & 0 \\ 0.707 & 0.6 & -1 \end{bmatrix}$, find the vector $\boldsymbol{x}$ using your algorithm!

2. Load this .mat file into your workspace. If you're on Python, use *Scipy* to load the file. This file contains the matrix $\mathbf{A}$ and vector $\boldsymbol{y}$ to be used. Apply your algorithm to find a sparse representation in 64 dimensions of the 32 dimensional measurements.

Task 2 :

In this Task, you will be working on 2D Images. Download the Files for this Task from here. The File *Task2.mat* contains an Image and a Matrix $\mathbf{A}$.

The Matrix consists of 256 Images[Size : 5x5] from the Discrete Cosine Transform (DCT) dictionary stacked upon each other. You should try and obtain the Sparse Vector for the Image(let's call this $\boldsymbol{x}$).

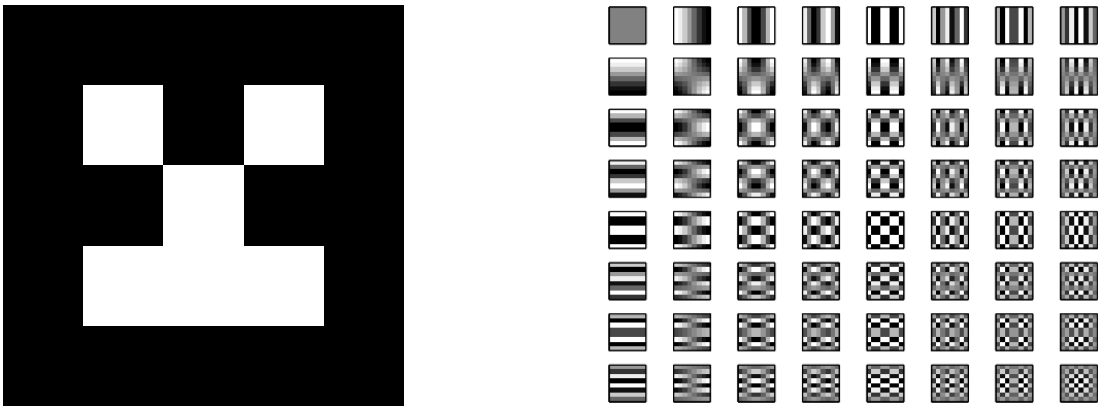

Figure 1: The Input Image and a 8x8 Section of a standard DCT Dictionary

Task 3 :

Now that you can Represent an Image using $\boldsymbol{x}$, Task 3 involves Feature Extraction from these images. Your aim is to perform 4 Types of Edge Feature Extraction. They are the First and Second order Edge Extraction in both Horizontal and Vertical Directions respectively. An Image has been provided to test your code.

Here are some sample outputs, which include the Image Profiles for the Original and 2 of the Derivatives.

Figure 2: Sample Image

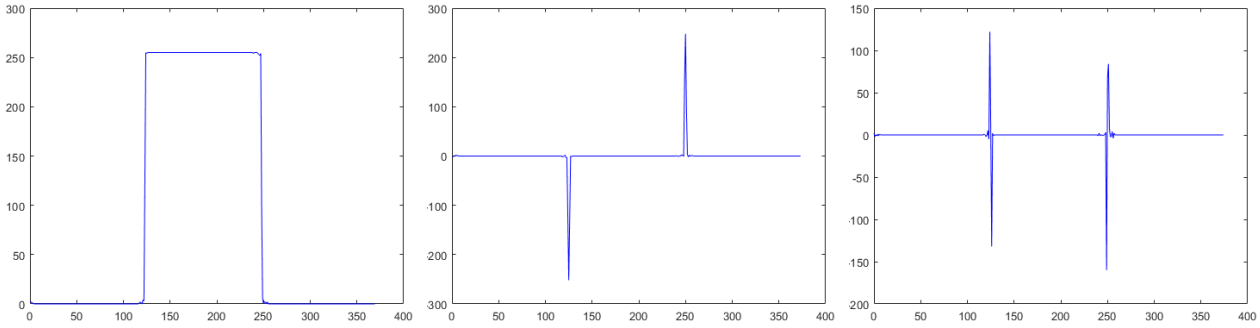

Figure 3: Horizontal Image Profiles for Sample Image and it's 1^{st} and 2^{nd} order Derivatives

Before we dive into the next Task, here is a brief overview of the whole process.

The solution involves the use of 2 Dictionaries : $\mathbf{D_l}$ and $\mathbf{D_h}$ which are essentially sets of Basis vectors for the Sparse Representation. The dimensions of the dictionaries are as follows :

$$\mathbf{D_l} : 100 \times 512 \text{ and } \mathbf{D_h} : 25 \times 512$$

Moving on, the final image you obtain will be the super resolved image of size 128×128 for a Scale Factor of 2. After obtaining the output, it will be checked against the ground truth for calculation of Signal to Noise Ratio (SNR).

Task 4 :

In this task, you will be given an Input Image of dimensions 64×64. Your objective is to upscale the Image by 2 using Interpolation.

Figure 4: Input Image : Cameraman

Task 5 :

In this task, Extract the 4 features given in the *Task 3* for the output image obtained in *Task 4*. Generate a sparse representation for every 5×5 patch using the dictionary $\mathbf{D_l}$.

Task 6 :

Now that you've successfully completed *Task 5*, use the Sparse vectors obtained in conjunction with the Dictionary $\mathbf{D_h}$ to obtain the Super Resoluted Image! If you've done it correctly, you should now have a 128×128 Super Resoluted version of the Input Image.

Task 7 :

For this Task, we have provided the Ground Truth Input Image. Your goal is to calculate the SNR of the Super Resoluted Image that you obtained in *Task 6* with respect to the given Ground Truth Image. The output should be a floating point number in dB.

Here are the outputs for a different Sample Case :

Figure 5: Ground Truth

Figure 6: Interpolation

Figure 7: Super Resolution

Task 8 :

Now that you have performed Super Resolution on Grayscale Images, you can move on to perform the Same Task on Color Images. For this task, we've provided a Test Image. You are expected to submit a Super Resolved version of the same.

Figure 8: Input Image

Here are the outputs for a different Sample Case :

Figure 9: Ground Truth

Figure 10: Interpolation

Figure 11: Super Resolution
