Proba V - Diary of an odyssey

Matteo Taccola

11/09/2019

Beginning of the Odyssey and a disclaimer

"Tell me about a complicated man.
Muse, tell me how he wandered and was lost…"

The Odyssey began 11 years ago when I was hired by ESA to work as Optical engineer on a mission considered by many at the edge of feasibility…. ProbaV

Disclaimer: This is my first experience with both image processing and machine learning. I apologize in advance for all stupid things/inaccuracies/wrong statements/puerilities that I can say…

Basic idea (most probably not ideal!)

1. For a specific image set we have several low resolution images and we need to provide a super resolved image.

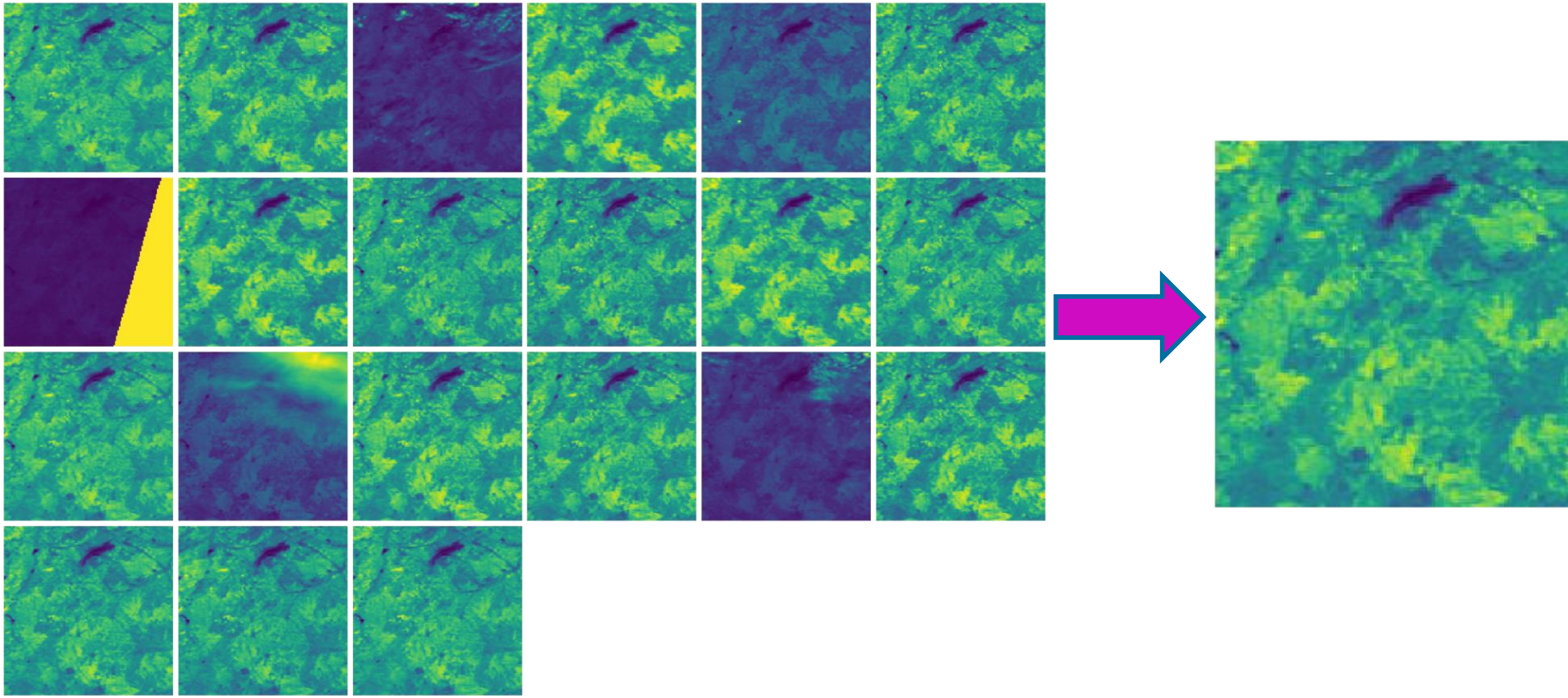
    Several issues: corrupted pixels, images are not perfectly coregistered, images are taken in different conditions/time, different number of images for different sets

2. Define a reference image for each set. This is used to coregister all the low resolution images and to replace corrupted pixels

3. Use classic image processing/enhancement to generate a super resolved image

4. Use convolutional neural networks to further improve the results (as last improvement)

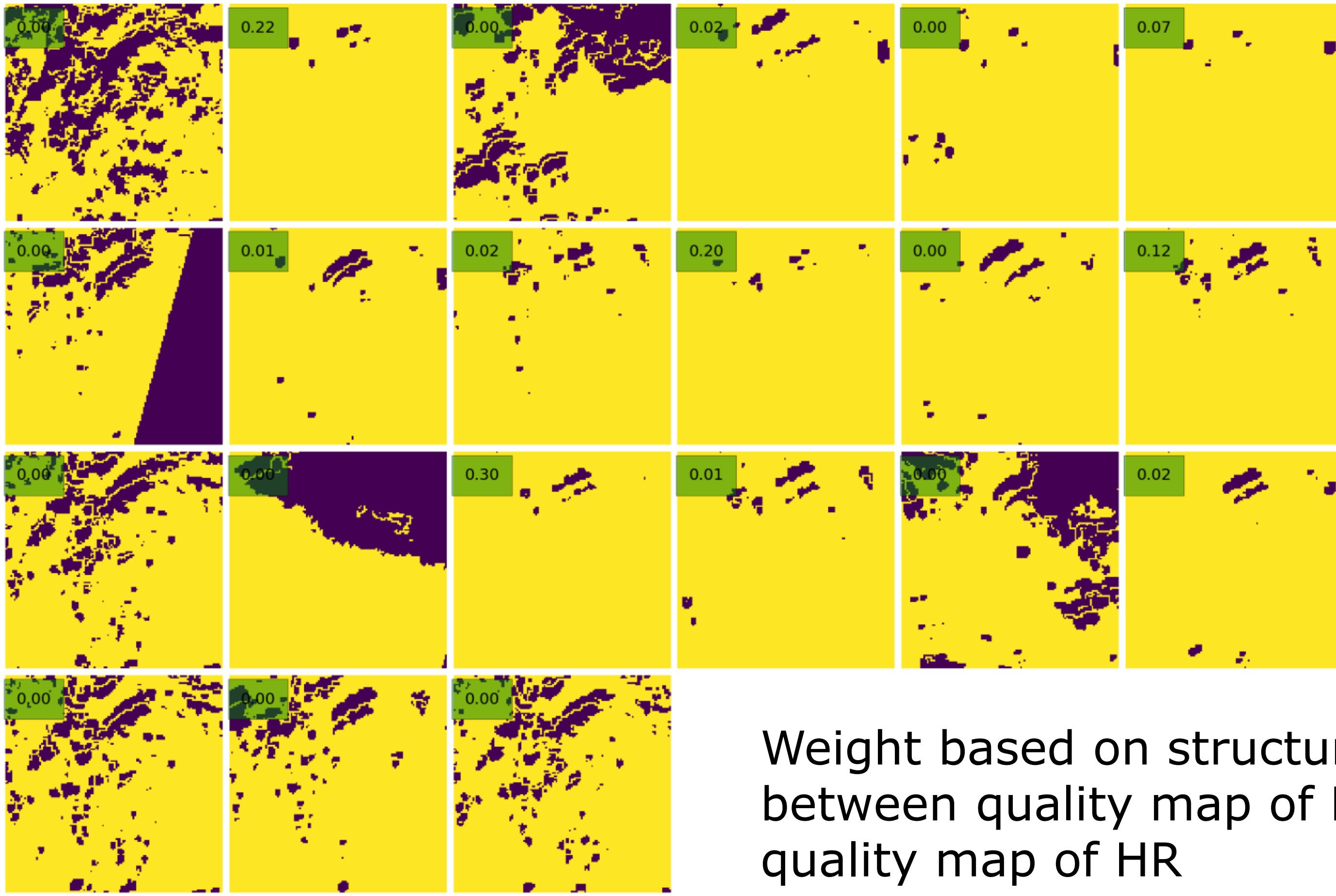
First step: create “average” LR image

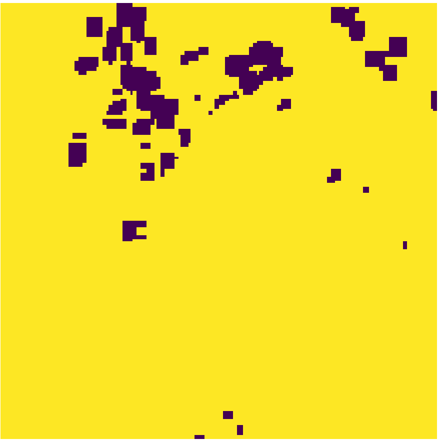


1. Remove “bad” pixels
   Flag and mask “bad” pixels (high intensity **and** flagged in QM maps)
   Perform average of masked image (ignoring bad pixels)
   If average image contains still bad pixels than substitute “bad” pixels with interpolation between neighbor pixels
2. Find proper average method

How to perform average?

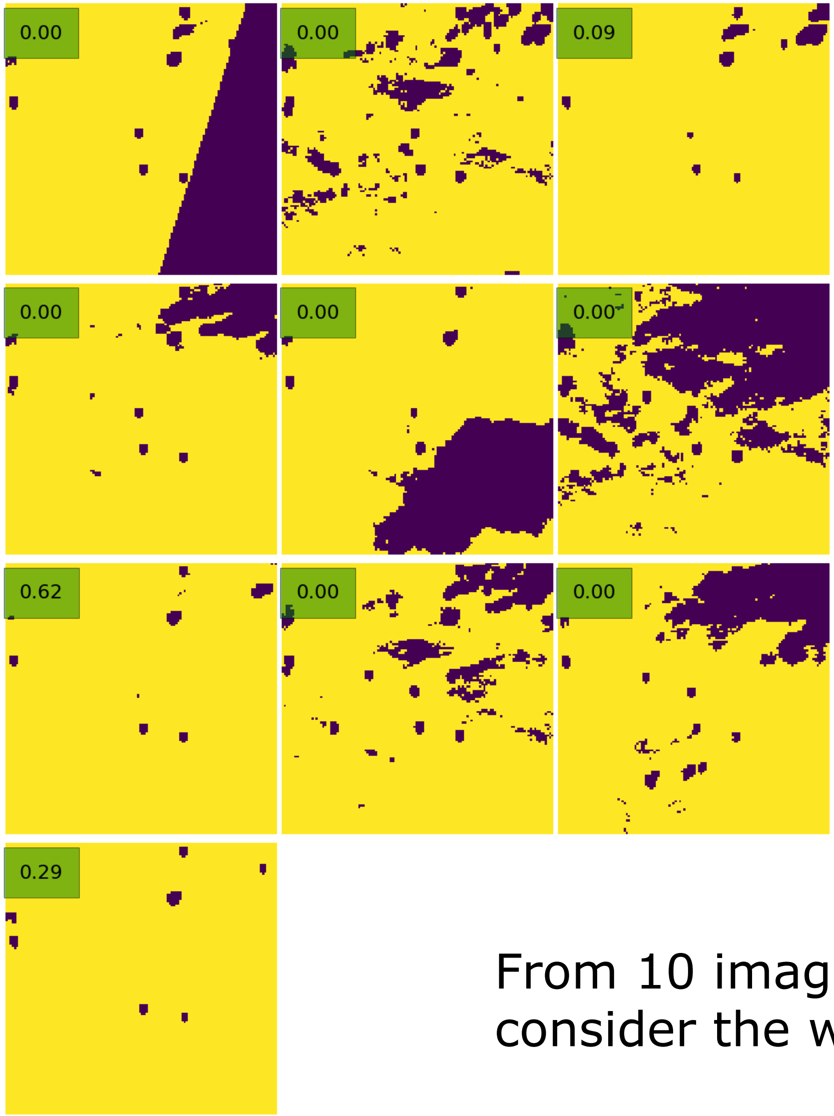
1. Same weight to each image (arithmetic average)
2. More importance to images with lower number of bad pixels
3. More importance to images with quality map close to HR quality map





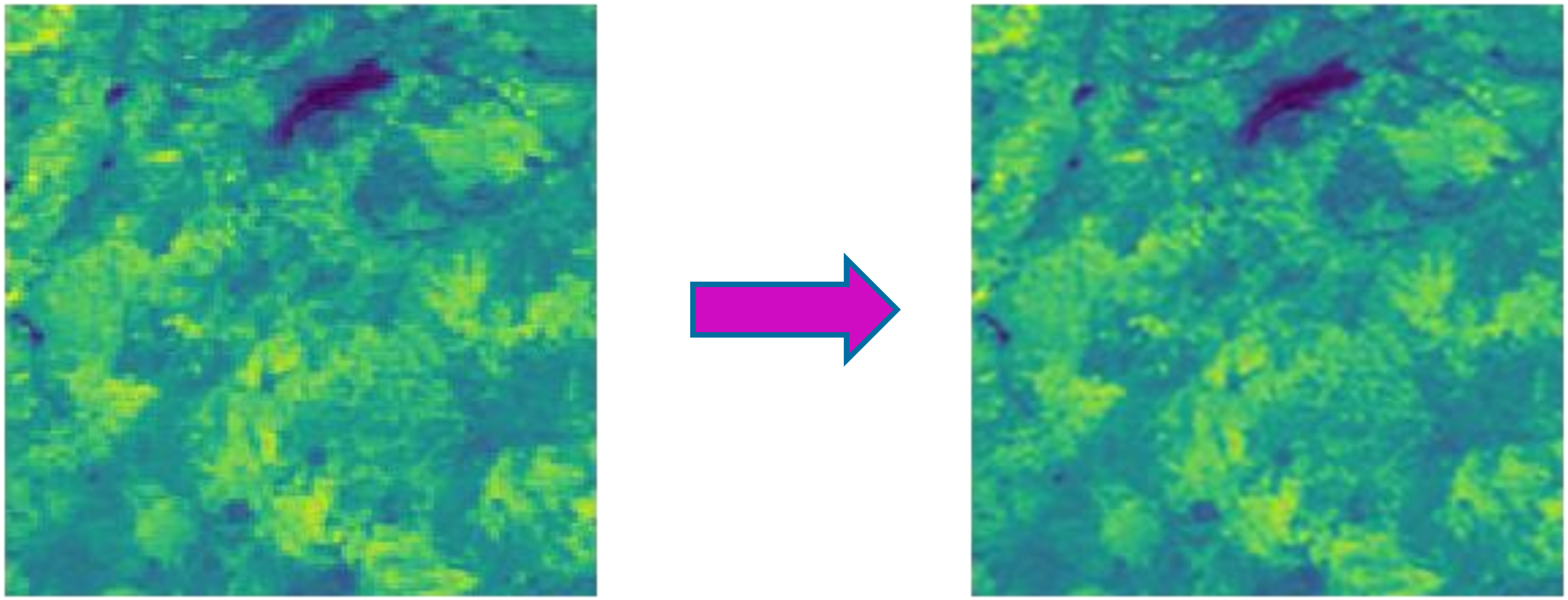
Weight based on structural similarity index between quality map of LR and downscaled quality map of HR

Another example (imgset0008)



From 10 images of the dataset I just consider the weighted average of 3

Second step: Upsampling



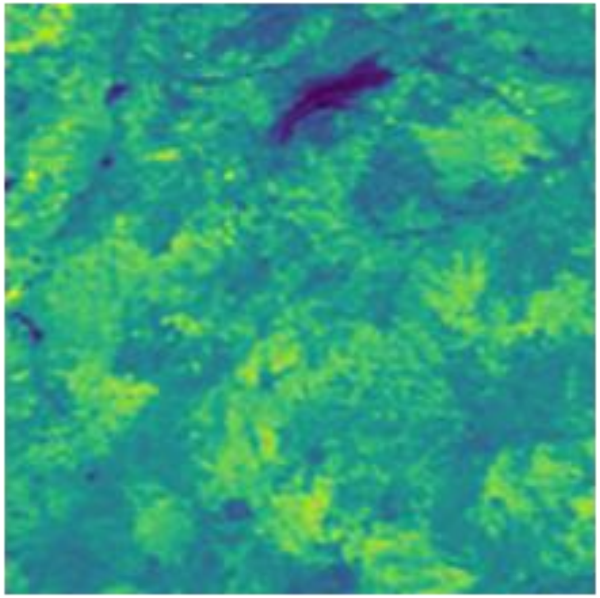
Tried different methods (bicubic, Lanczos, 5th order interpolation..)

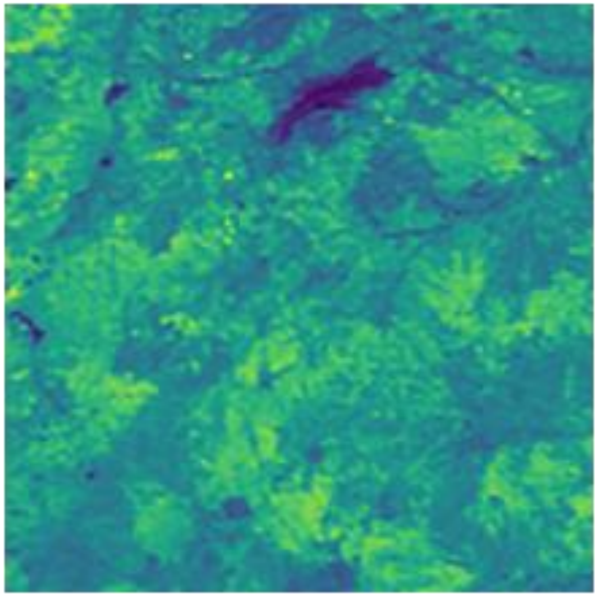
5th order interpolation gives the best results but introduce artefacts

Baseline is Lanczos (ref: https://en.wikipedia.org/wiki/Lanczos_resampling)

Note: According to Wikipedia Laczos is "considered the "best compromise" among several simple filters for this purpose" → I agree!

Third step: Image sharpening





My understanding (post competition) is that convolutional neural networks can better replace all this ....

This is my reference average image

All classical image processing methods

- unsharp mask (both gaussian and laplacian)
- adaptive filter

Ref: https://homepages.inf.ed.ac.uk/rbf/HIPR2/unsharp.htm

- histogram normalization

My final digital filter depends on few (8) parameters

- All parameters are obtained from minimization of score function of train images (Nelder – Mead simplex algorithm)

4th step: Coregistration

All the LR images are coregistered but within a certain accuracy

It is important to improve coregistration with respect to the reference SR image

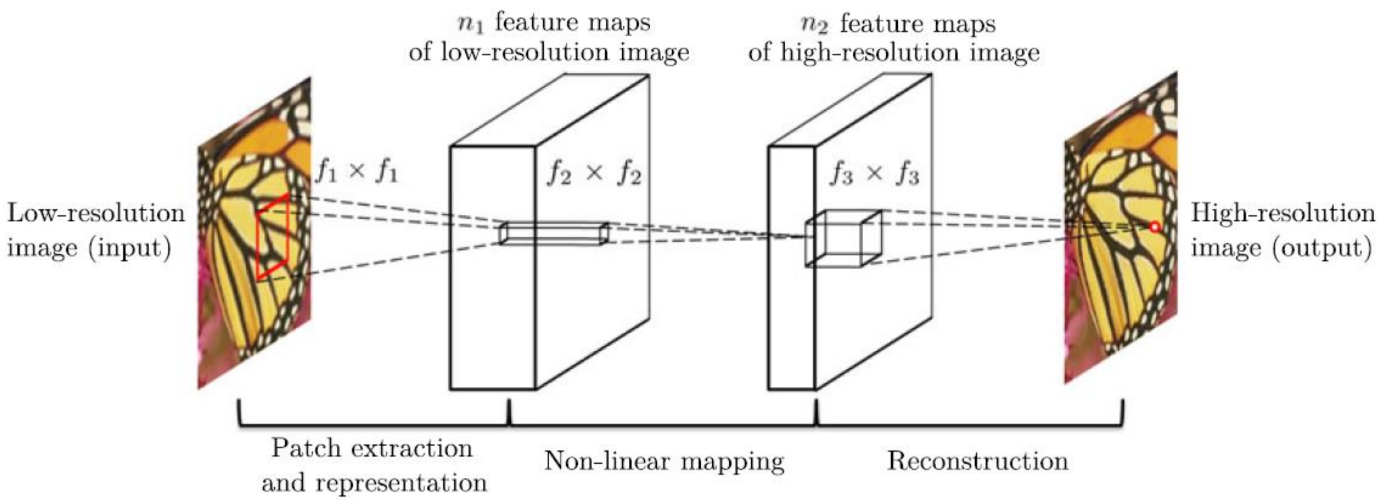
Take all LR images (for a given scene) → Upsample (Lanczos) → Substitute bad pixels with corresponding pixels of reference image → Image sharpening of each LR image → Compute shifts reference image vs each LR → Coregister LR images/ Average

skimage.feature register_translation to compute shifts
scipy.ndimage shift to coregister

5th step: Deep learning

I computed SR images for all **1160** training scenes

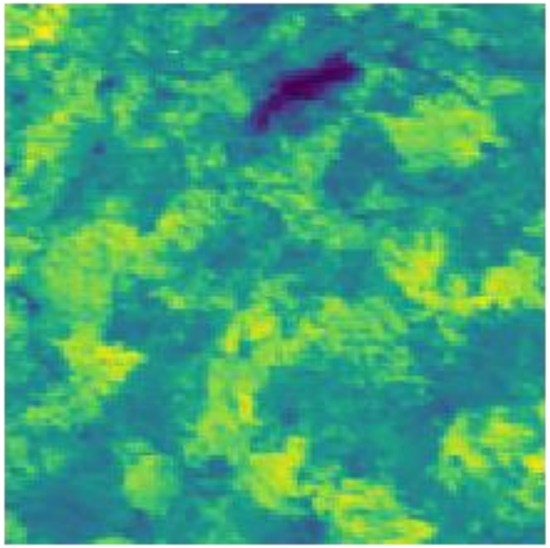
I used these images to train a Super-Resolution Convolutional Neural Network (SRCNN*)



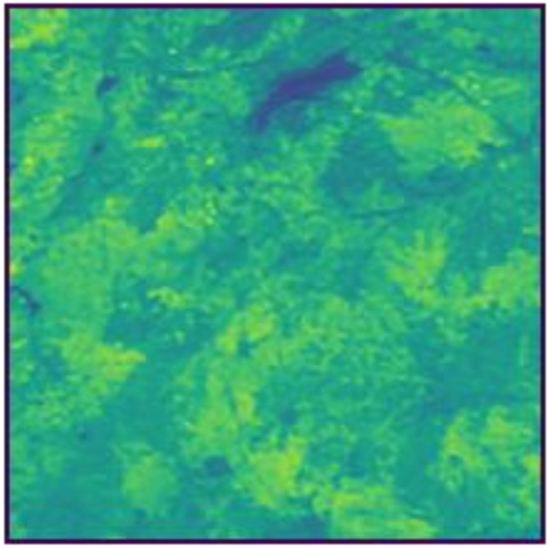
*reference: ***Image Super-Resolution Using Deep Convolutional Networks***

Chao Dong, Chen Change Loy, Kaiming He, Xiaoou Tang

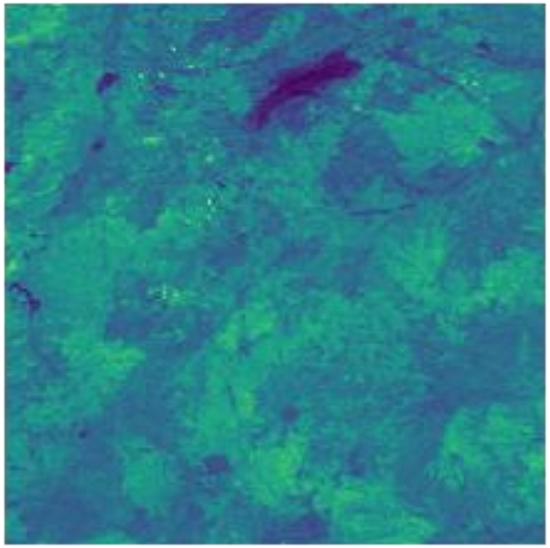
Results



one LR image



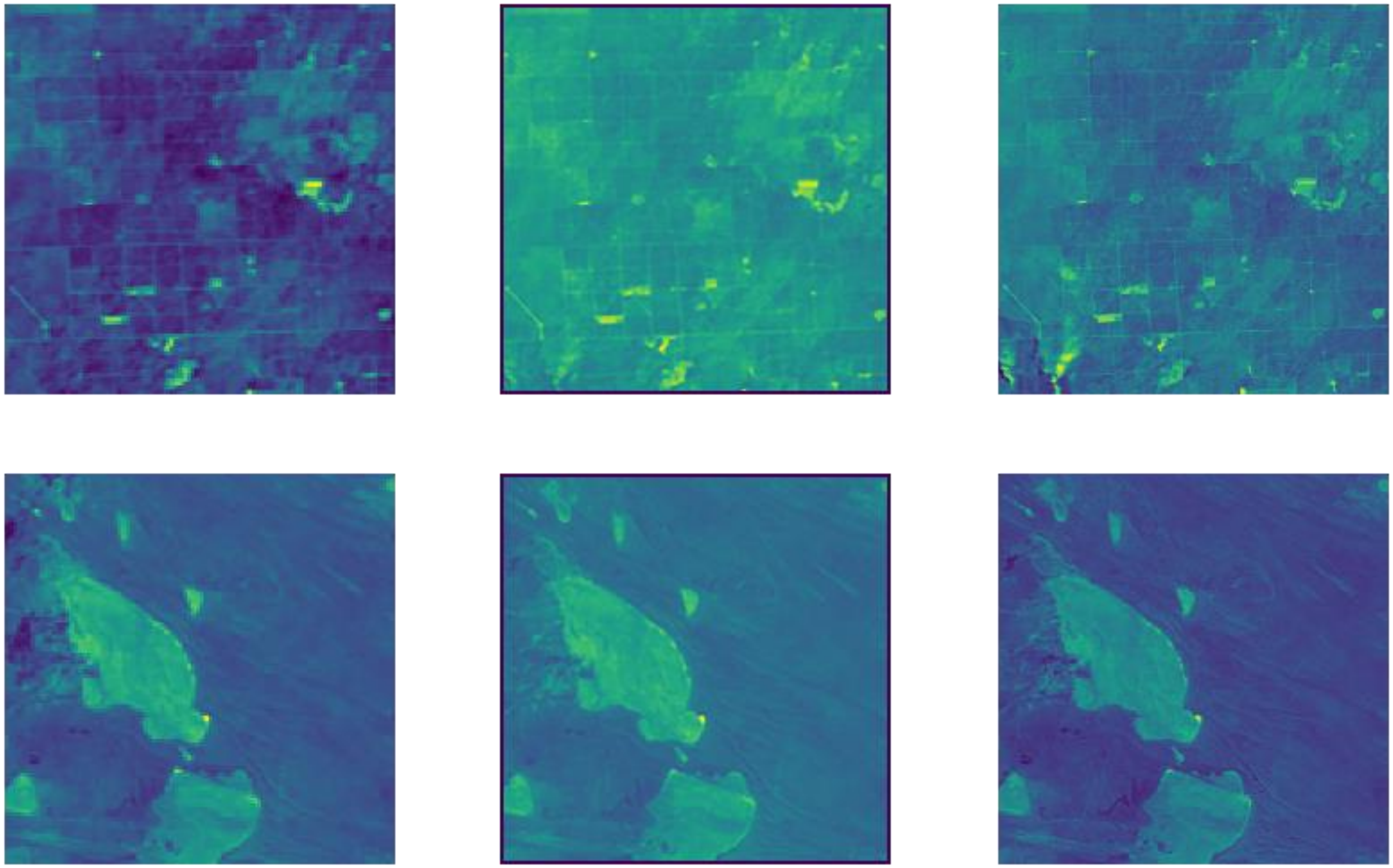
My SR image



HR image

3 pictures are worth 3000 words …

More results…



Possible improvements

1. Define a better average method/ranking between LR images possibly using machine learning

   Also use Dice coefficient instead of Structural Similarity Index (more appropriate for masks)

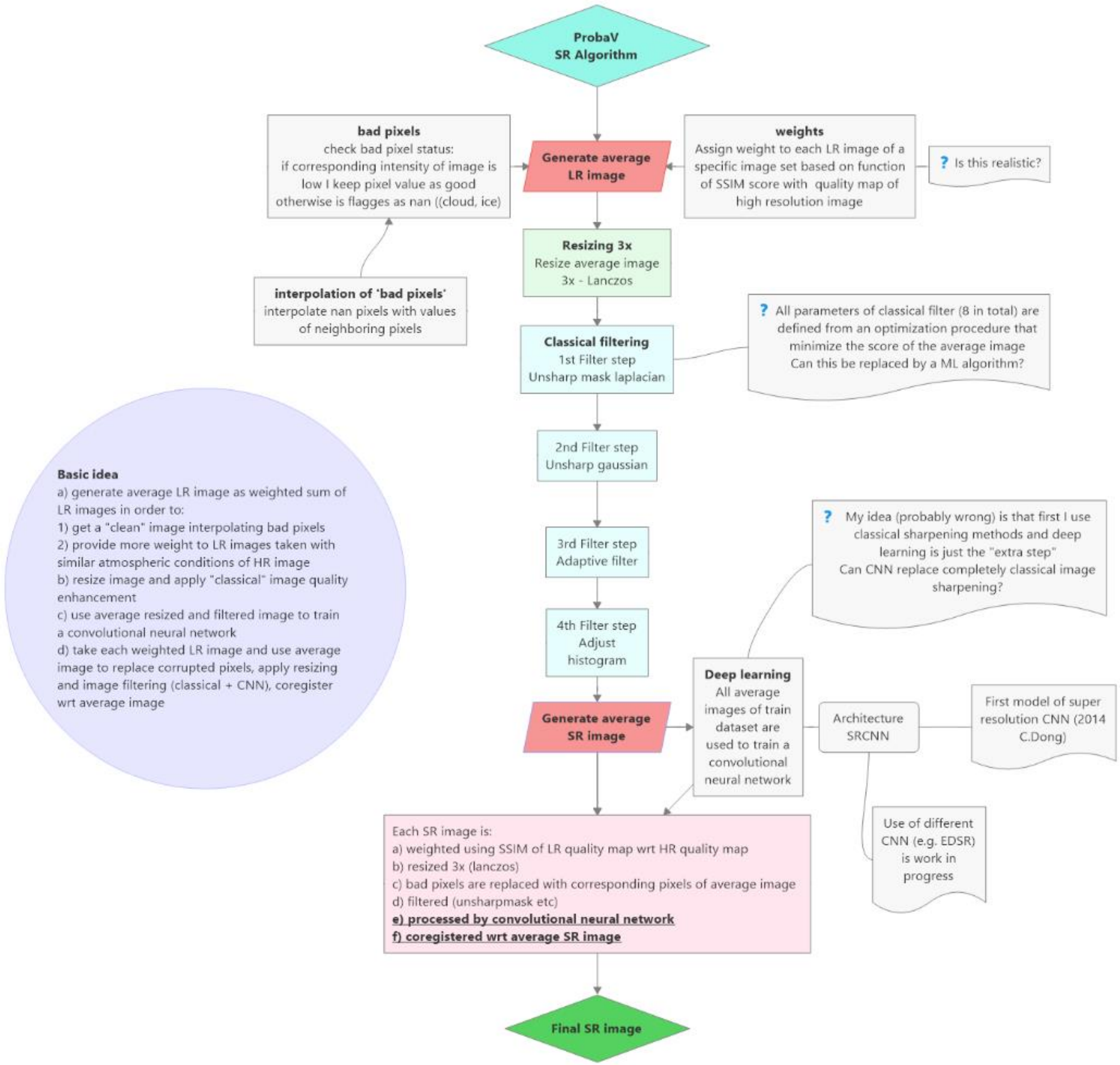
2. Use CNN instead of classic image processing for computing the reference SR image

   I should be able to obtain better results than unsharp mask or other digital filters…

3. Different network architectures for single image super resolution should give better results than SRCNN (e.g. EDSR) but I was not able to follow this path
   - Resource limitations (google colab is a nice (**and free!!**) GPU resource but certainly not ideal)
   - Time constraints

4. Use image augmentations for training (horizontal and vertical flip, rotations)

Summary flow chart



Super resolution – Few notes

- Combining several low resolution images help to improve spatial resolution

  (first obvious statement)

- I got one of the best improvements making use of high resolution quality map
  - Provide ranking/weights of different images

- The best way to obtain a given spatial resolution is to build a dedicated payload

  (second obvious statement)

  This imply high cost/complexities

- My main open question is: which additional parameters/measurements/data possibly coming from other instruments can be used to improve spatial resolution?

End of the Odyssey

At the end of this adventure ProbaV is still orbiting and I am still lost ….

…but I learned a lot about python programming, machine learning and image processing and I really had a lot of fun (and now I have also a new passion for competitions!!).

Special thanks to the Advanced Concept Team group for organizing such a nice competition.

Special thanks to Luis Simoes for providing such a great python module for this competition (and also for all interesting discussions after the competition).

Special thanks to my wife because I worked (too much) during weekends, holidays, evenings, nights etc etc..

Thank you!!!