Machine Learning applied to Planetary Sciences

PTYS 595B/495B

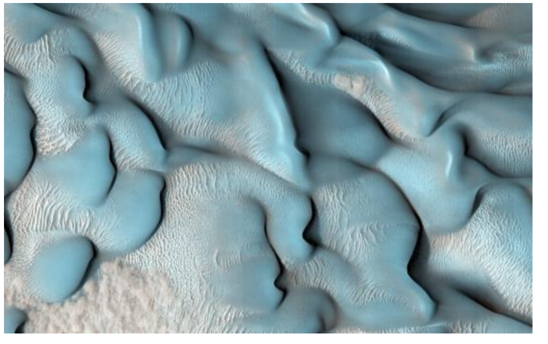
Leon Palafox

https://leonpalafox.github.io/MLClass/

Application to Landform Detection

Problem Statement

- Different communities are interested in different landforms:



Problem Statement

- The easiest alternative is to manually tag everything you can find during your lifespan.
  - People actually do this
- Second easiest:
  - Pay an undergrad to do it.
- The cheapest alternative is to create an automated landform detection algorithm.

Dataset

- Data sets in planetary science are ugly:
  - Large sized images
  - Few Labels
  - Unbalanced Datasets (More negative examples than positive examples)
  - Formats that are usually foreign to most of the Computer Vision community. (JP2, GeoTiff, IMG)

Datasets

- To create a good output product, we need to know the necessities of the community.
  - Non-georefferenced images are cute, but at the end useless.
  - Geotags adds some level of complexity

Dataset

- Usually they are not available to us:
  - We need to create a tool that manually extracts at least a couple hundred examples.
    - Negatives and Positives
  - At the same time we can start adding some flexibility to the algorithm.

Algorithm Selection

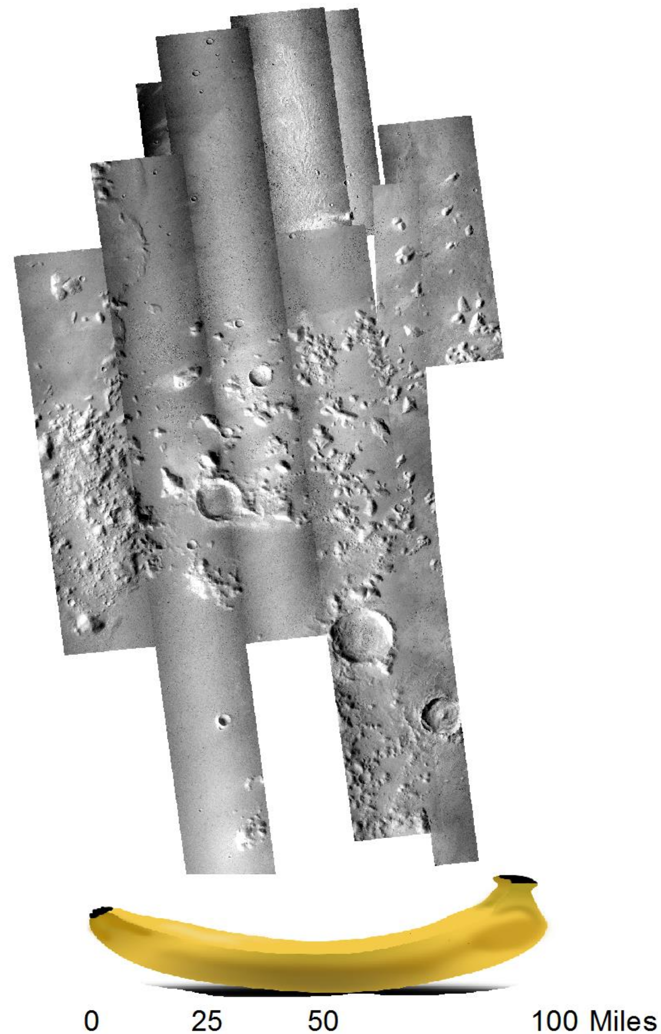
- A good approach would try more than one algorithm:
  - No free lunch
  - We need to test and optimize (CV, ROC) each algorithm for its different variations.
- We also need to take into account speed of the algorithm.

Overall Structure

- Problem: Detect Landforms (Volcanic Rootless Cones)

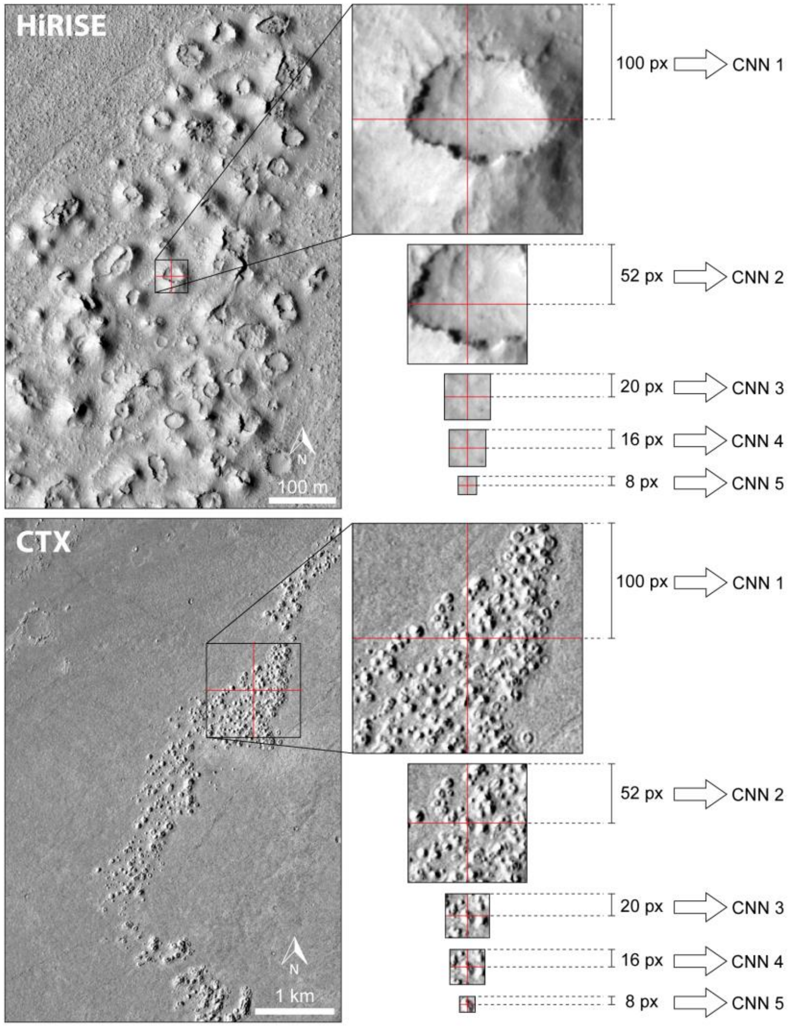
Problem

- Find the VRCs in a large area
- They can have very small diameters.
- Images have to be geotagged.
- Images are very High Resolution.



Dataset

- HiRISE and CTX images (instruments on board of the Mars Reconnaissance Orbiter)
- Very unbalanced dataset.
- We extract 4 examples out of each manual tag.
- We ran five classification schemes in parallel.
- We ran the histogram normalization in all the images.



Dataset

- Why did we extract 4 examples:
  - If we only target the dead center image, the algorithm will generate very limited maps.
  - We artificially increase our training dataset.
- Why five classifiers:
  - Since we are using both HiRISE and CTX, the scales vary.
  - Cones are different sizes.
  - We expect that small window classifiers detect smaller cones better.

Algorithm selection

- We pit an SVM against a CNN:
  - For the SVM we had to use a preprocessing technique called Histogram of Gradients (HoG)
  - HoG is good for Shadows and scales

Meet Lenna, and yes she is a 1972 playmate.

Algorithm selections

- CNNs are incredibly good and don’t need any sort of preprocessing
- For both SVMs and CNNs, we used CV and Grid Search (SVM).
  - Testing dataset of 30%
  - CNN: Tested different size of input windows, number of layers, activation units, regularization.
  - SVM: Kernel, Kernel parameters, HoG parameters.

Training running time

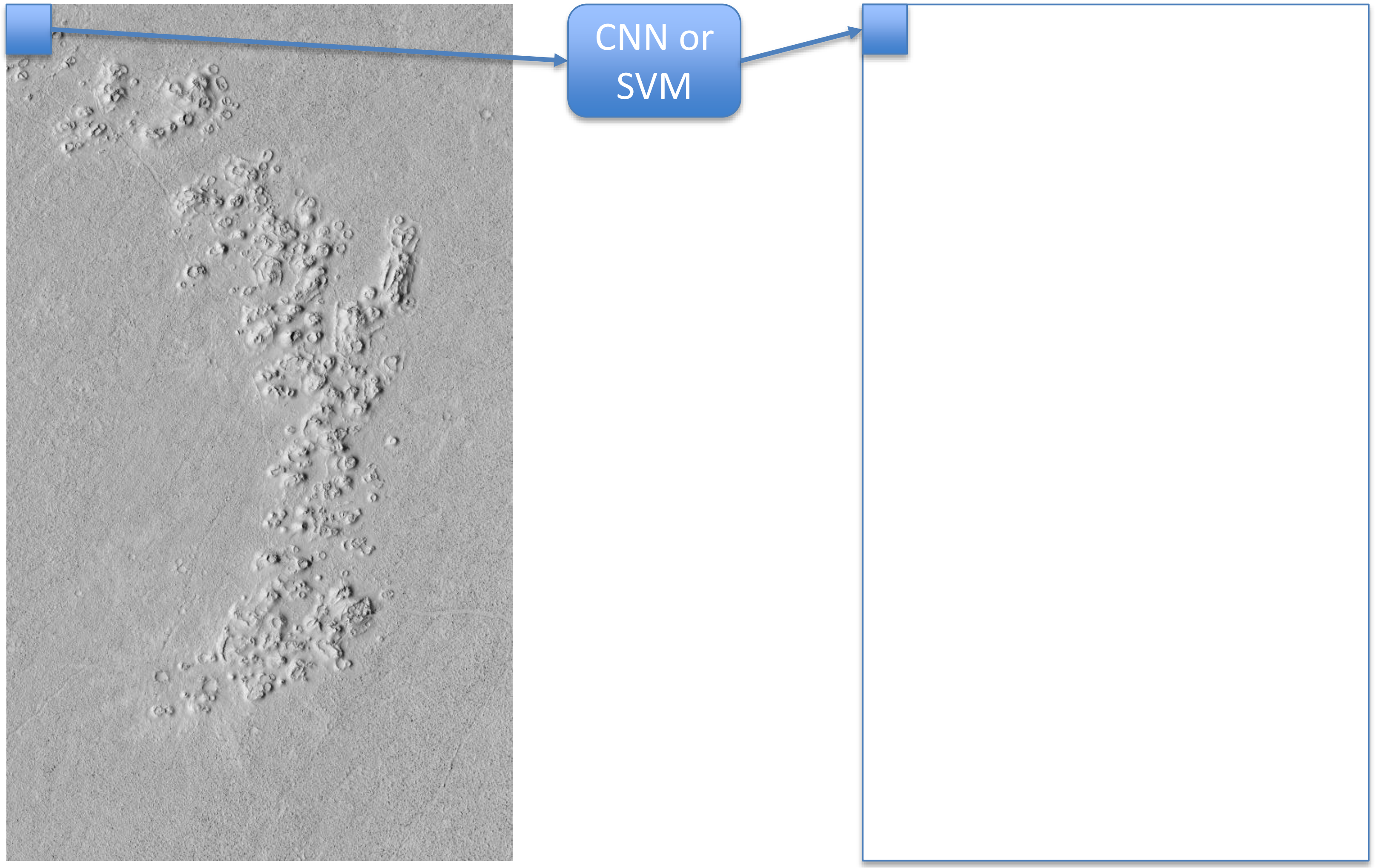
- SVMs: Training and CV takes about 3 hours.
- CNNs: Training and CV takes about 48 hours.

- CNNs have more parameters than SVMs, and their training algorithm also takes more.
- But in Remote Sensing we care about classification times.

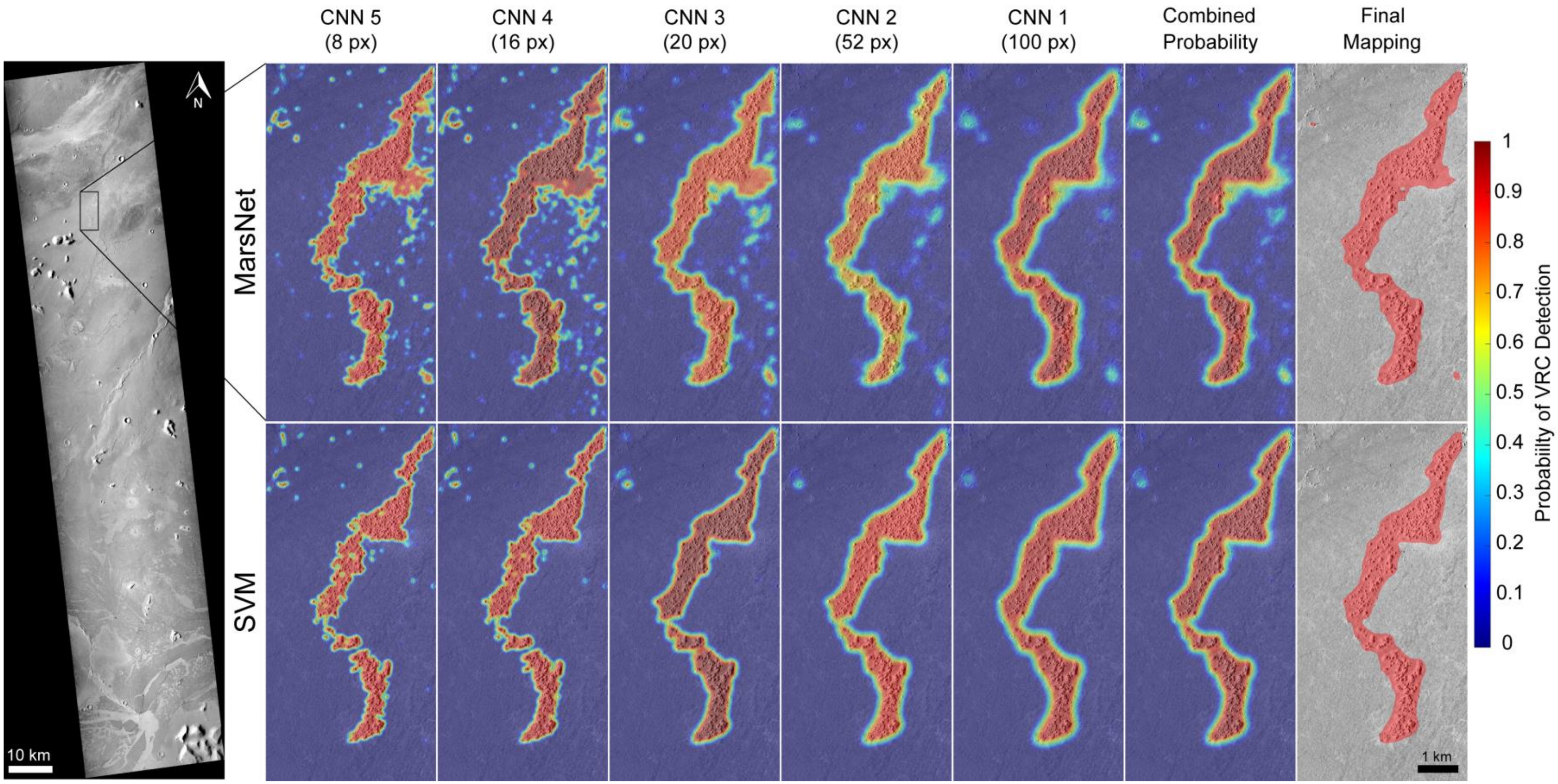
Full Pipeline

- Get Image
- Extract georeferenced information
- Run Classification pipeline
  - SVM or CNN
- Save the image
- Give it to the expert (hopefully easily available)
- Wash and Rinse again

Classification pipeline



Final Mapping



Final Mapping