# Week 9

Mining Asset Detection (MAD)

# Satellite images - Hopefully for the very last time

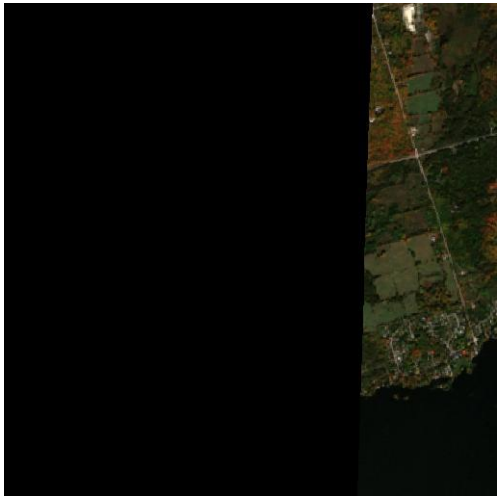
In order to increase the quality of the satellite images, I tried to download unprocessed GEE images

Result:

- Increase in quality for some
- Unexpected issues with swath of Satellites (should be harmonized?)

Fix?

- Do an on site quality assessment (fits into the squeeze phase)



# Satellite images - Larger image download

In order to add minor augmentations to the image

- Rotation
- Translation
- Zoom

We need bigger images to create these augmentations, but…

**GEE has max image download size of 32MB…**

# Satellite images - Larger image download

We can download images in parts, let’s calculate (excluding metadata):

512 x 512 pixels, with 13 bands, each being a 16bit integer: **6MB** for a 5km x 5km image

Increases quadratically => 10km x 10km already at **24MB**

# Satellite images - Larger image download

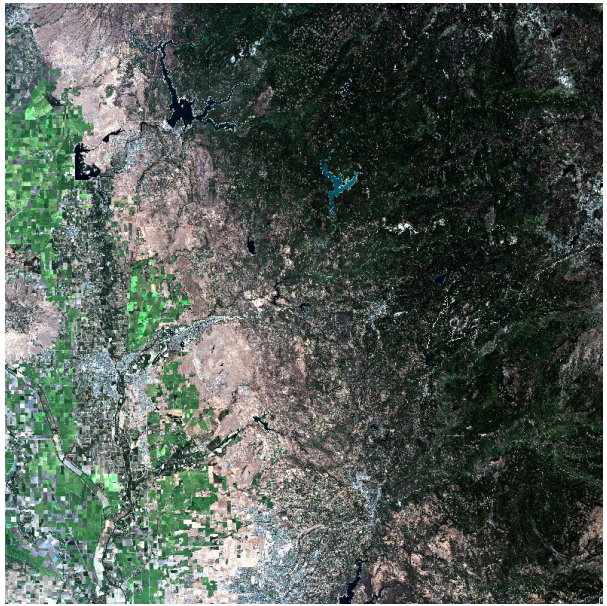
Possible quick fixes:

- Create the augmentation on site and only download final image (should be very slow for a whole dataset due to requirement to download the image)
- Download each band separately, single band ~ 0.5 MB for 5km => 40km x 40km maximally possible image

# Satellite images - AWS Sentinel 2

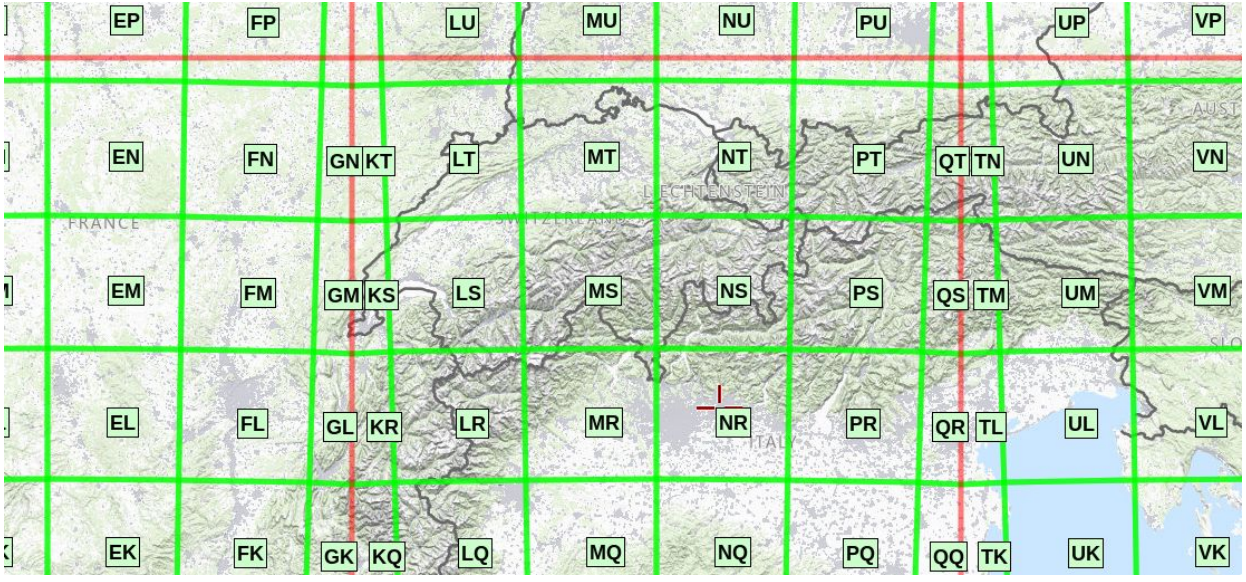
The very surprising fix:

- Sentinel 2 data can be accessed publicly given this documentation
- Every single image is 100km x 100km
- Download speeds in the 600Mbits! (4x GEE)
- Images look fantastic



# Satellite images - AWS Sentinel 2 - the caveat

Their system uses MGRS, the worst “projection” I had to work with yet



# Satellite images - AWS Sentinel 2 - the caveat

Since we're not relying on GEE quality control anymore, weird things can happen

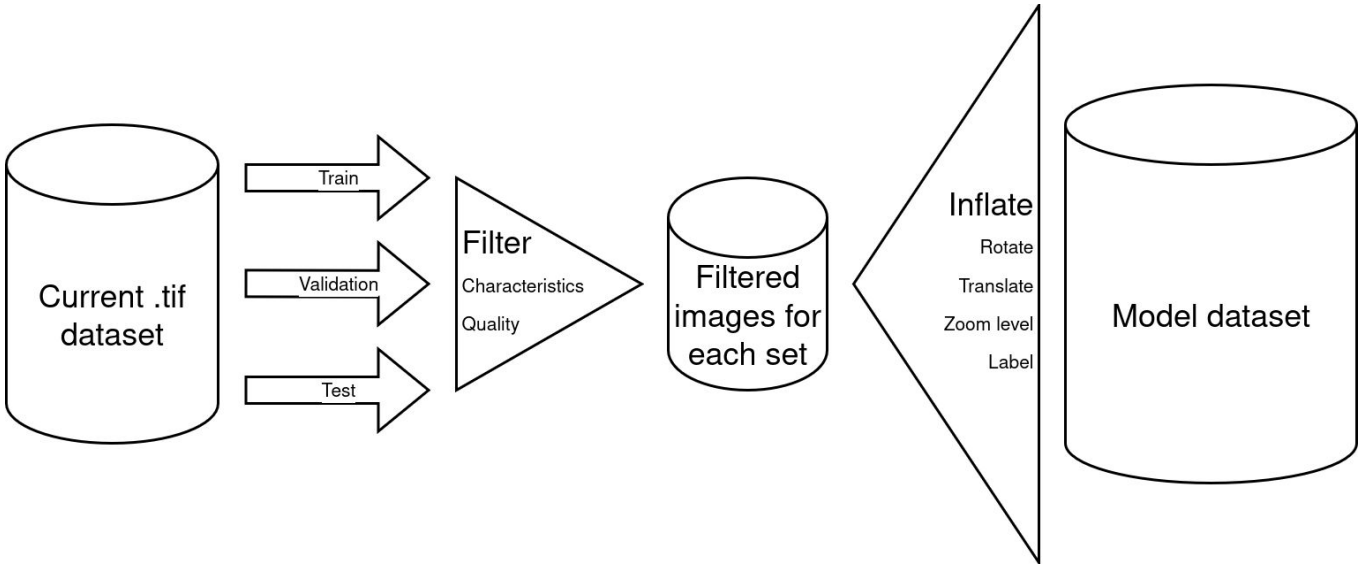
- Unexpected cloud coverage
- Glitched images (Faults happened during download or reconstruction of images)
- Lower quality than expected in images

Fix: Based on metadata (all cached on our side) poll different images with better quality metrics

**Could even be automated!**

After iterations of improvements we should have only quality satellite images to work with.

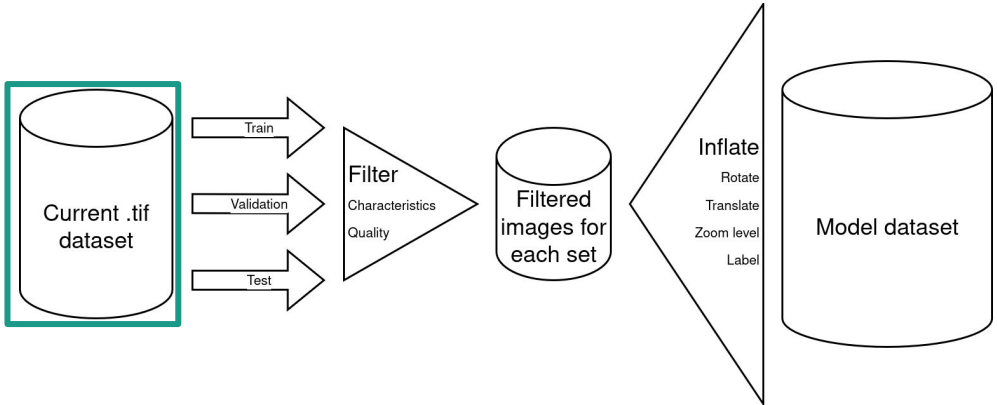
# What now? - Dataset pipeline



# What now? - Dataset pipeline - Origin data

.tif dataset is frozen for a model creation.

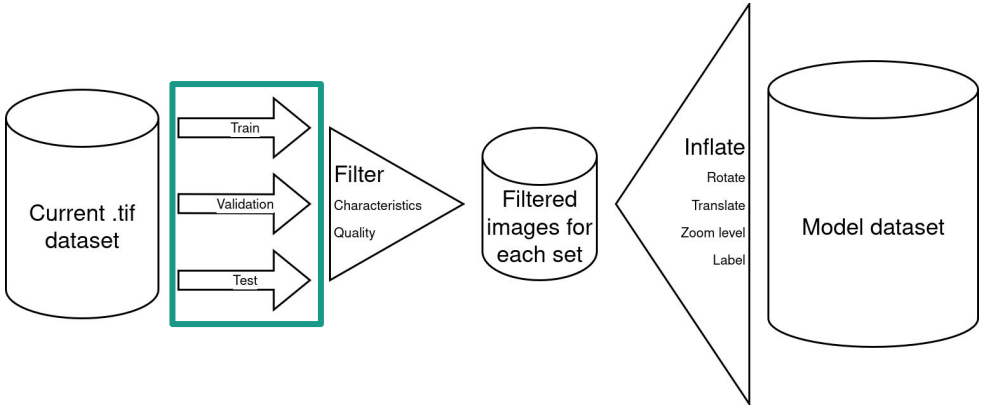
- Based on filtering results (such as no data due to swath) the dataset may be refined over time.
- General extent stays the same.



# What now? - Dataset pipeline - Split

Split done before any filtering or expansion.

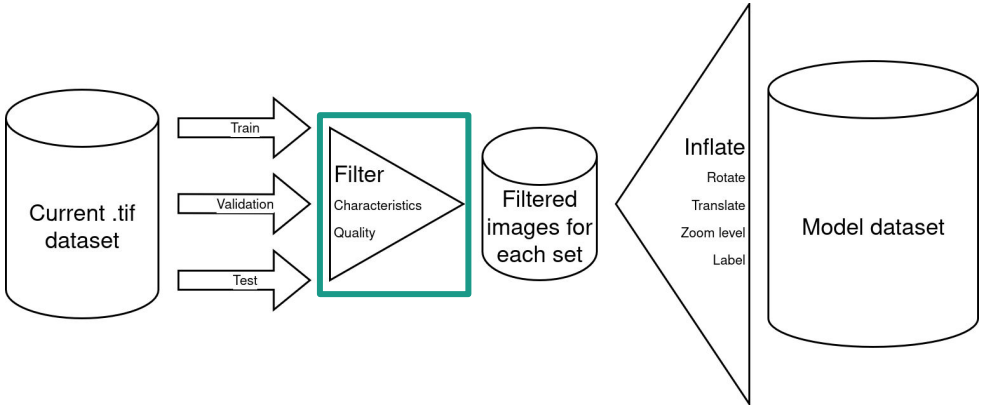
- More manageable (~5000 bare satellite images)
- Ensures no matter the expansion, that no information leaks into the test set
- Ensures comparability between models



# What now? - Dataset pipeline - Filter

Filters incoming .tif images based on quality and characteristic

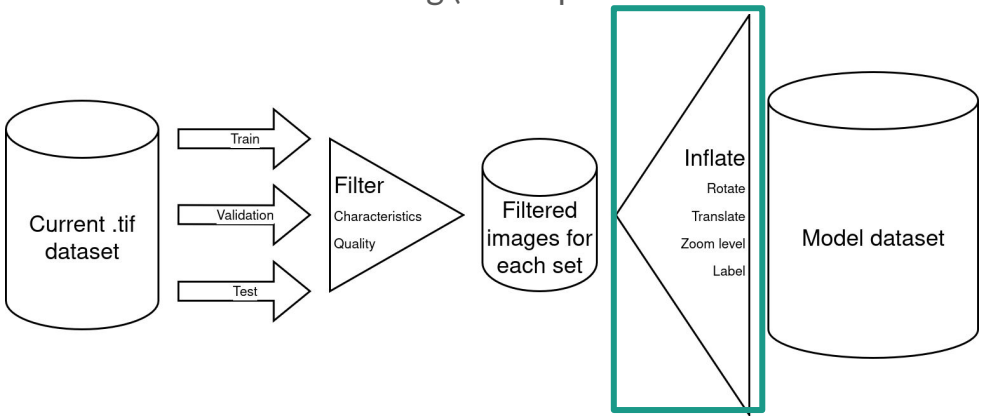
- Quality, to form a feedback loop with the original dataset
- Characteristic, to allow special model training like ecoregion specific models.



# What now? - Dataset pipeline - Inflation

Inflates the .tif image into its .jpg variants

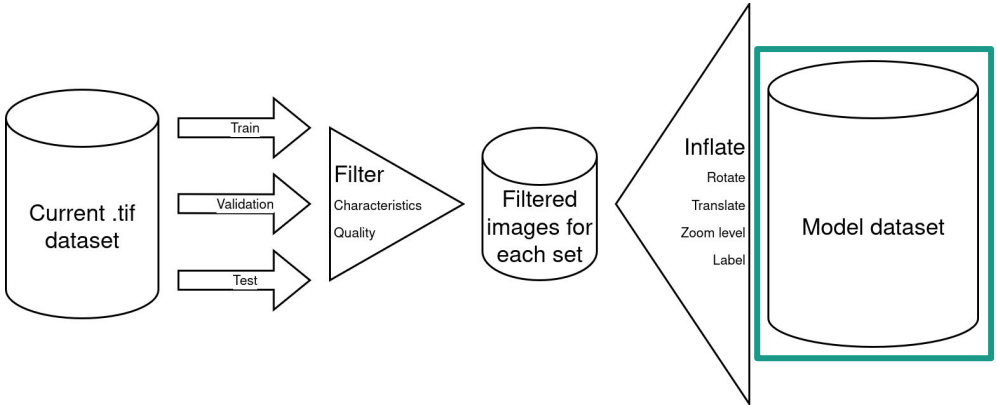
- Rasterizes the .tif and outputs the rasterized images (Essentially as we've done until now)
- Creates random permutations of the images, such as rotation, translation etc.
- Also has to do the labeling (as the permutations make this a bit more tricky)



# What now? - Dataset pipeline - Model dataset

Final dataset over which to perform the model training (either lasso or yolo)

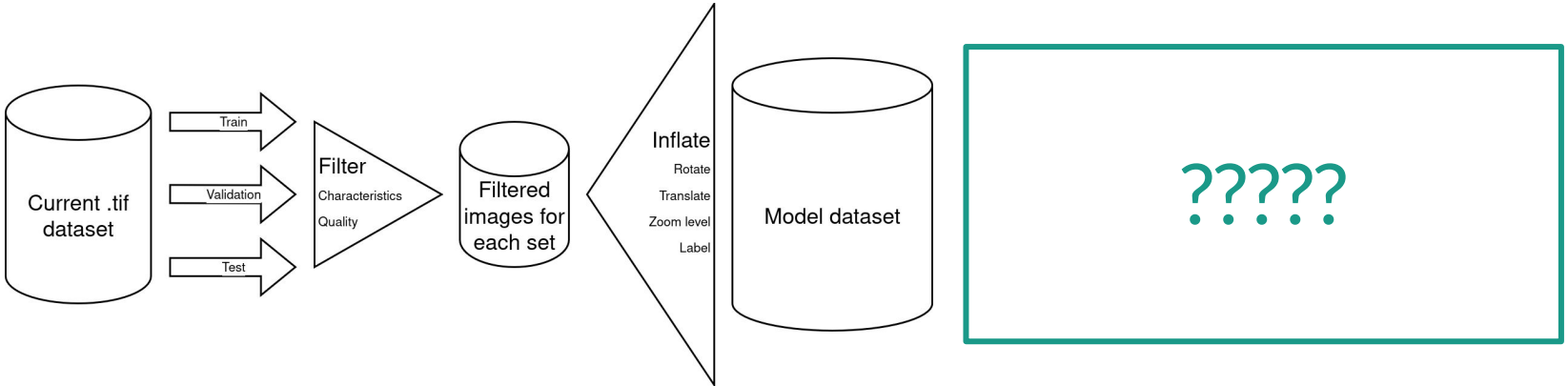
- Must fit the model requirements for the input.



# What now? - Model training

Question for later:

- How to design the exact model training (specifically yolo)
  - Choosing parameters
  - Which YOLO model to take



# What now? - Model comparison

The separated dataset can be used to uniformly test the performance of all resulting models

- Allows for a final comparison
- Most cost effective:
  - Use YOLO's inbuilt metrics and prediction on a test set
  - Augment it with our own (only per image)