

SUMMER SCHOOL GI_SALZBURG 2022

GLACIER MAPPING USING CNN

Presented by Arioluwa Aribisala, Sofia Delgado,
Saad Ahmed Jamal, Oscar Narvaez, Sarah Poch,
Zhibek Solpieva



CONTENT

- Introduction
- Study Area and Data
- Basic concepts
- Flowchart
- Final results

INTRODUCTION

GLACIAR MAPPING

WHY IS IT IMPORTANT?



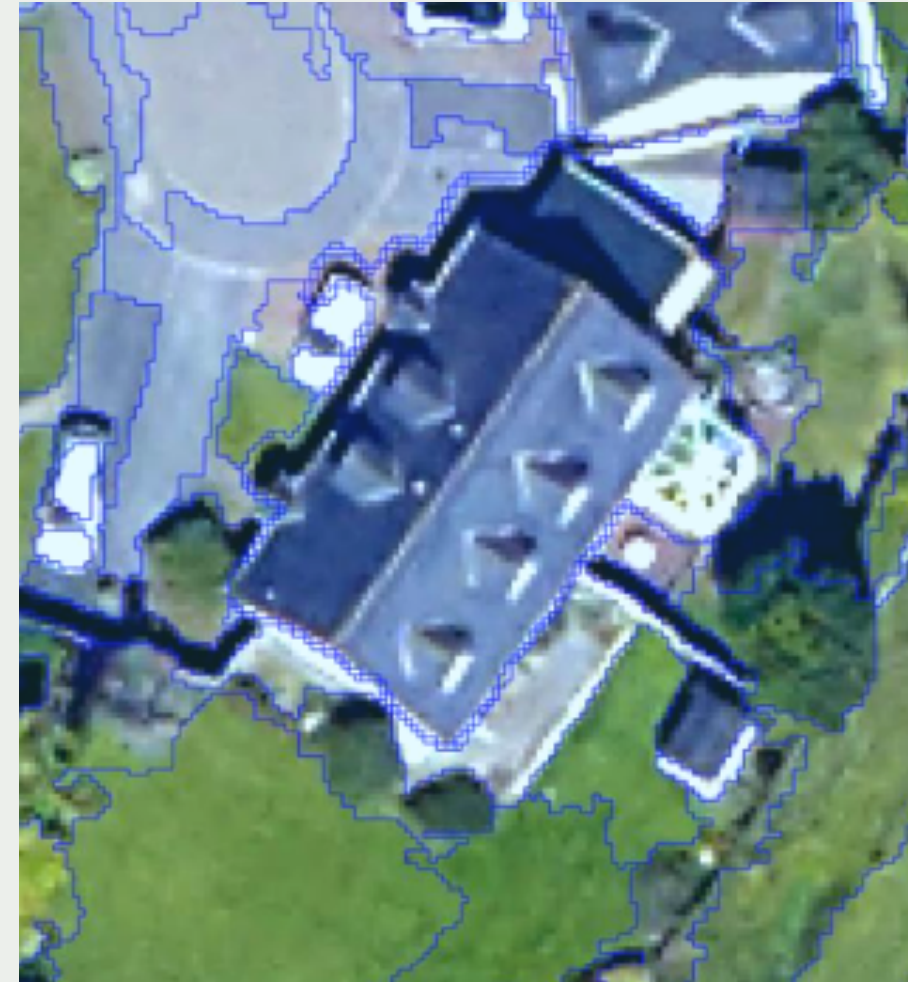
Solheimajokull, Iceland in 2007 and 2015.



BASIC CONCEPTS



DEEP LEARNING
AND
CONVOLUTIONAL
NEURAL
NETWORKS



OBJECT-BASED
IMAGE ANALYSIS

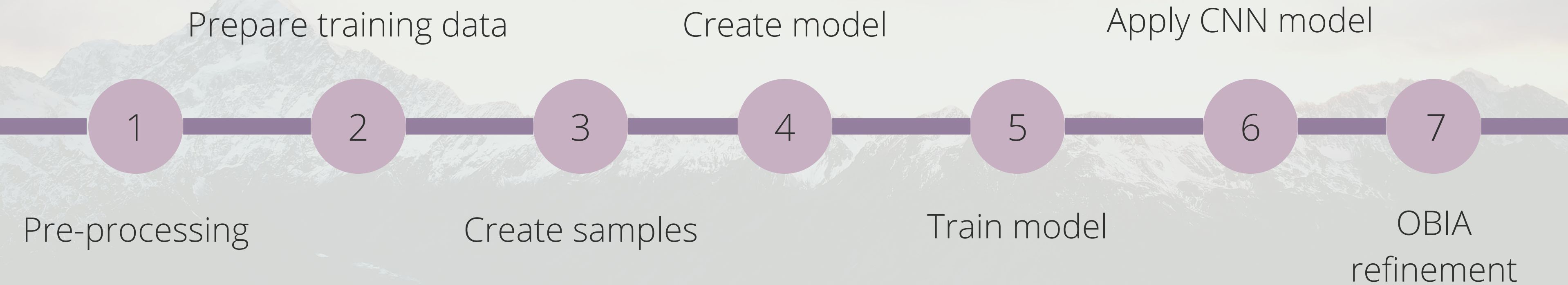
STUDY AREA AND DATA

THE GLACIERS IS LOCATED IN HUNZA VALLEY IN PAKISTAN

- Sentinel 2 optical satellite imagery from 04.08.2018.
- Coherence data based on two Sentinel 1 images (05.08.2018 and 17.08.2018).
- Topographic data from the ALOS Global DEM.
- Software eCognition



FLOW CHART



1 Pre-processing

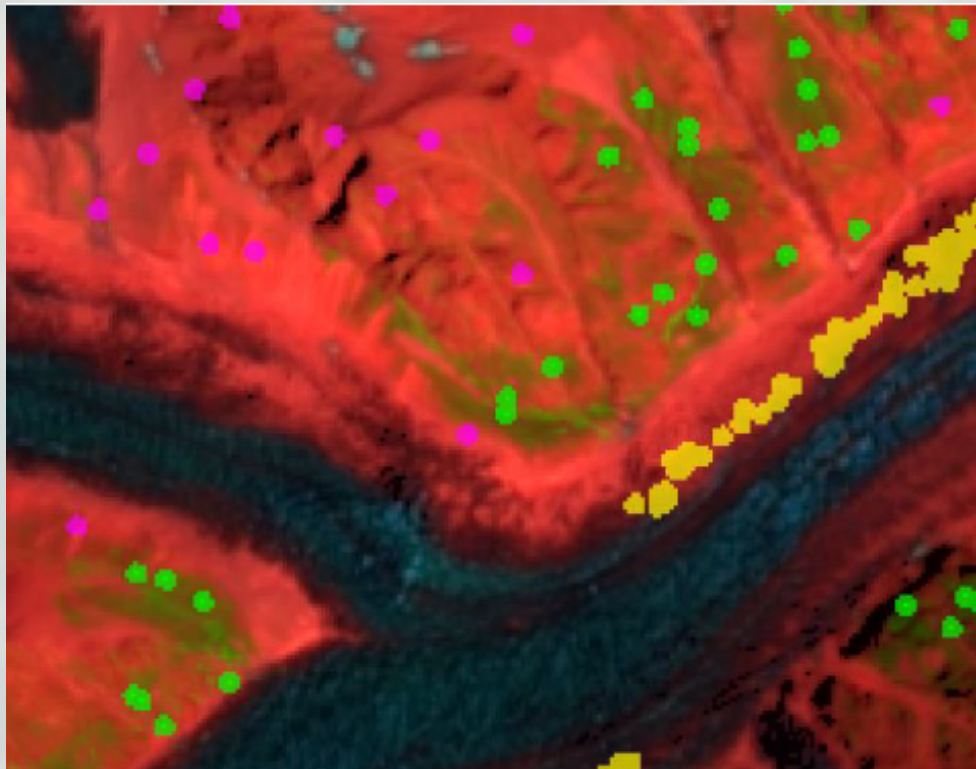
- For each pixel, it makes one segment
- These steps are creating normalized values for each raster band one by one
- The CNN algorithm takes input in 32 bit floating point type
- Layer arithmetics calculated

```
Create CNN
├── create layers
│   ├── set custom view settings on first pane
│   └── 16:42.078 layer preparation
│       ├── 16:42.078 convert to 32-bit binary
│       │   ├── 08.609 chess board: 1 creating 'New Level'
│       │   ├── 03.828 layer arithmetics (val 1.001, layer dem range [9000,1e+30] to layer dem_nulled[32Bit float ])
│       │   ├── 38.922 at New Level: green_max = max(Mean green)
│       │   ├── 37.422 at New Level: blue_max = max(Mean blue)
│       │   ├── 42.719 at New Level: red_max = max(Mean red)
│       │   ├── 56.578 at New Level: nir_max = max(Mean nir)
│       │   ├── 01:23.797 at New Level: swir_max = max(Mean swir)
│       │   ├── 01:38.187 at New Level: dem_max = max(Mean dem_nulled)
│       │   ├── 02:13.547 at New Level: slope_max = max(Mean slope)
│       │   ├── 02:48.703 at New Level: sar_coherence_max = max(Mean sar_coherence)
│       │   ├── 01:20.281 layer arithmetics (val "blue/blue_max", layer blue_32[32Bit float ])
│       │   ├── 30.688 layer arithmetics (val "green/green_max", layer green_32[32Bit float ])
│       │   ├── 18.109 layer arithmetics (val "red/red_max", layer red_32[32Bit float ])
│       │   ├── 26.359 layer arithmetics (val "nir/nir_max", layer nir_32[32Bit float ])
│       │   ├── 10.219 layer arithmetics (val "swir/swir_max", layer swir_32[32Bit float ])
│       │   ├── 06.781 layer arithmetics (val "(dem_nulled/dem_max)", layer dem_32[32Bit float ])
│       │   ├── 06.532 layer arithmetics (val "slope/slope_max", layer slope_32[32Bit float ])
│       │   ├── 10.937 layer arithmetics (val "sar_coherence/sar_coherence_max", layer sar_coherence_32[32Bit float ])
│       │   ├── 02:19.750 delete 'New Level'
│       │   └── 0.094 delete image layer 'dem_nulled'
```

2

Prepare training data

- Creating a buffer around each thematic class



```
load in training data
53.094 prepare training data
  <0.001s buffer_size = 25
  <0.001s vector buffering/shrinking 'clean_ice' -> 'clean_ice_buffered' (delta=buffer_size, round)
  <0.001s vector buffering/shrinking 'debris_covered_glaciers' -> 'debris_covered_glaciers_buffered' (delta=buffer_size, round)
  <0.001s vector buffering/shrinking 'stable_slopes' -> 'stable_slopes_buffered' (delta=buffer_size, round)
  <0.001s vector buffering/shrinking 'lakes' -> 'lakes_buffered' (delta=buffer_size, round)
  <0.001s vector buffering/shrinking 'vegetation' -> 'vegetation_buffered' (delta=buffer_size, round)
  04.985 chess board: 1000000 creating 'level_1'
  0.922 unclassified with Num. of overlap: debris_covered_glaciers > 0 at level_1: debris covered ice
  0.937 unclassified with Num. of overlap: clean_ice_buffered > 0 at level_1: clean ice glacier
  0.922 unclassified with Num. of overlap: lakes > 0 at level_1: lakes
  0.922 unclassified with Num. of overlap: stable_slopes > 0 at level_1: stable_slope
  0.953 unclassified with Num. of overlap: vegetation_buffered > 0 at level_1: vegetation
  20.641 unclassified at level_1: chess board: 1
  21.765 unclassified with Mean nir < 900 at level_1: shadows
  01.047 clean ice glacier, debris covered ice, lakes, shadows, stable_slope, vegetation at level_1: chess board: 1
```


CNN

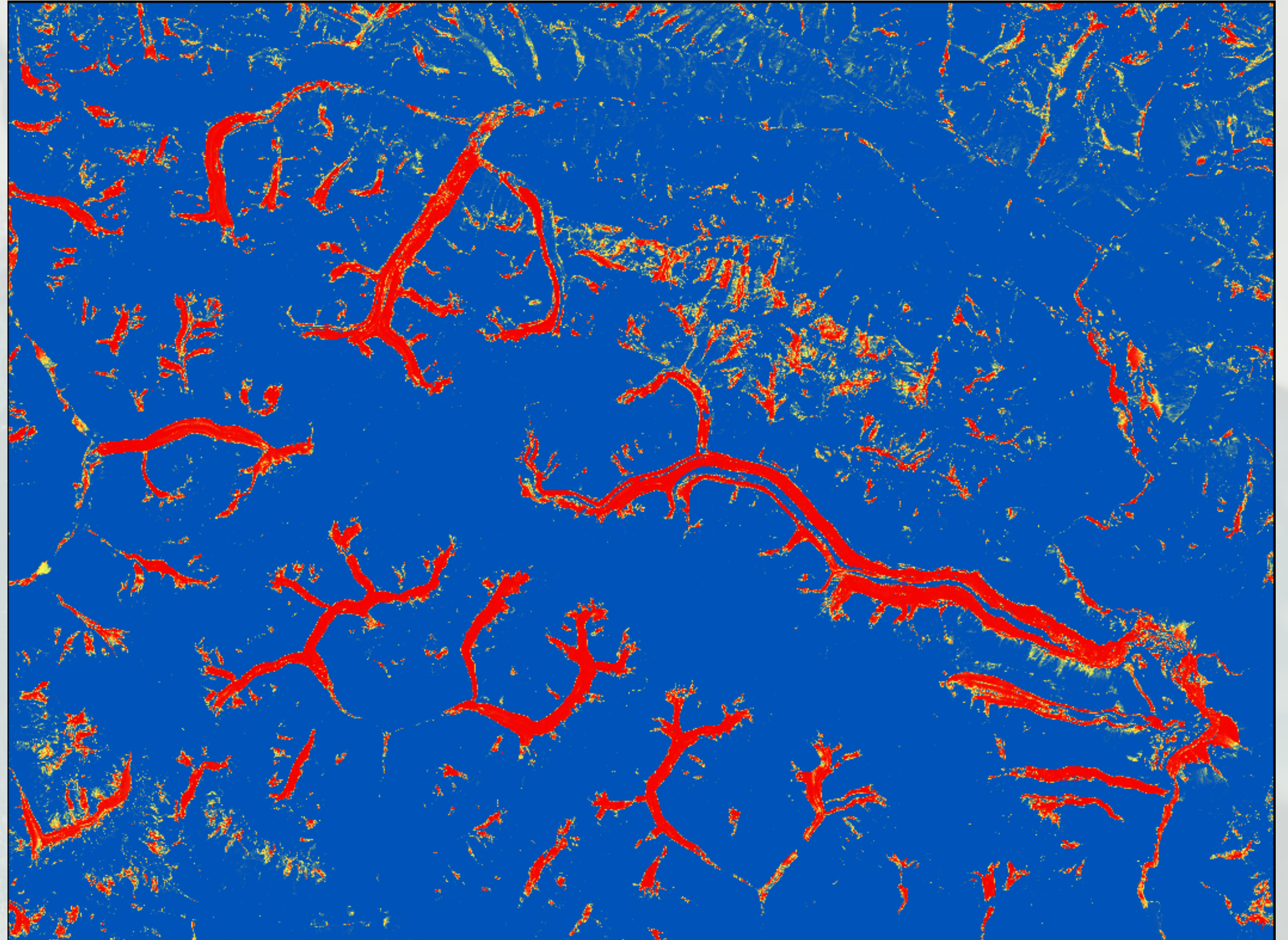


classes

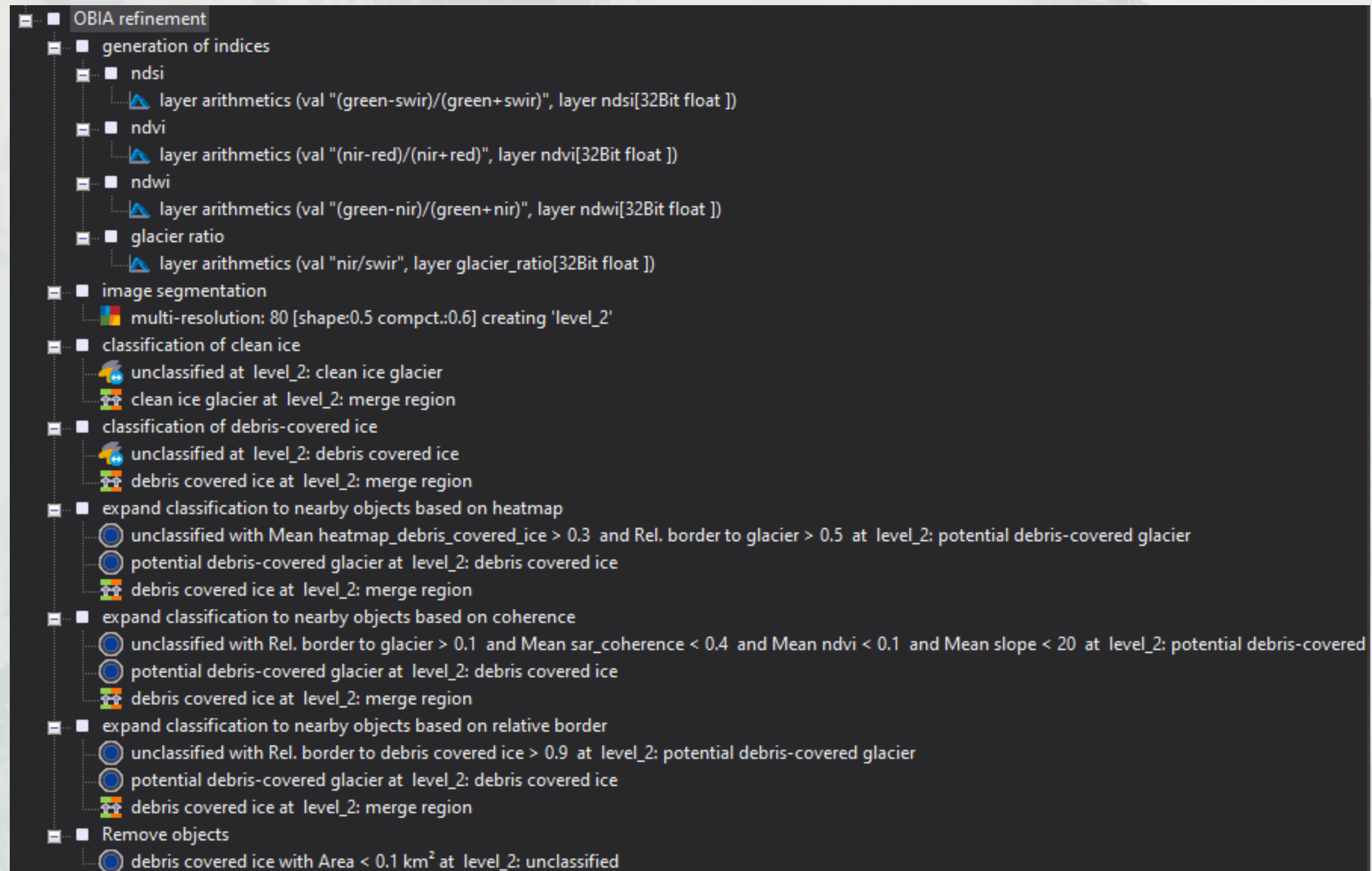
- glacier
 - clean ice glacier
 - debris covered ice
 - potential debris-covered glacier
- lakes
- shadows
- stable_slope
- vegetation

```
create samples
├── create
│   ├── sample_size = 16
│   ├── sample_folder = "(:Workspc.OutputRoot)\samples\"
│   ├── debris covered ice at level_1: generate 8000 raw sample patches sample_size x sample_size x 6 using layers ("sample_folder")
│   ├── clean ice glacier at level_1: generate 2000 raw sample patches sample_size x sample_size x 6 using layers ("sample_folder")
│   ├── lakes at level_1: generate 2000 raw sample patches sample_size x sample_size x 6 using layers (sample_folder)
│   ├── stable_slope at level_1: generate 18000 raw sample patches sample_size x sample_size x 6 using layers (sample_folder)
│   ├── vegetation at level_1: generate 2000 raw sample patches sample_size x sample_size x 6 using layers (sample_folder)
│   └── shadows at level_1: generate 2000 raw sample patches sample_size x sample_size x 6 using layers (sample_folder)
├── shuffle
│   ├── shuffle labeled sample patches ("sample_folder")
│   └── split labeled sample space from "(:Workspc.OutputRoot)\samples" to "(:Workspc.OutputRoot)\samples\samples1", "(:Workspc.OutputRoot)\samples\samples2"
├── 03:10.984 create, train, apply
├── 0.531 create model
│   └── 0.531 create 6-layer convolutional neural network with kernels [3,3,3,3,3], features[50,30,20,20,16] and pooling [false,false,false,false,true]
├── 03:10.453 train and apply model
│   ├── 01:09.765 train convolutional neural network (learn rate 0.0006) with 5000x16 samples ("(:Workspc.OutputRoot)\samples\samples1")
│   ├── 01:55.719 apply convolutional neural network (Class[debris covered ice]->Layer[heatmap_debris_covered_ice])
│   ├── <0.001s set custom view settings on second pane
│   └── 04.969 delete 'level_1'
```

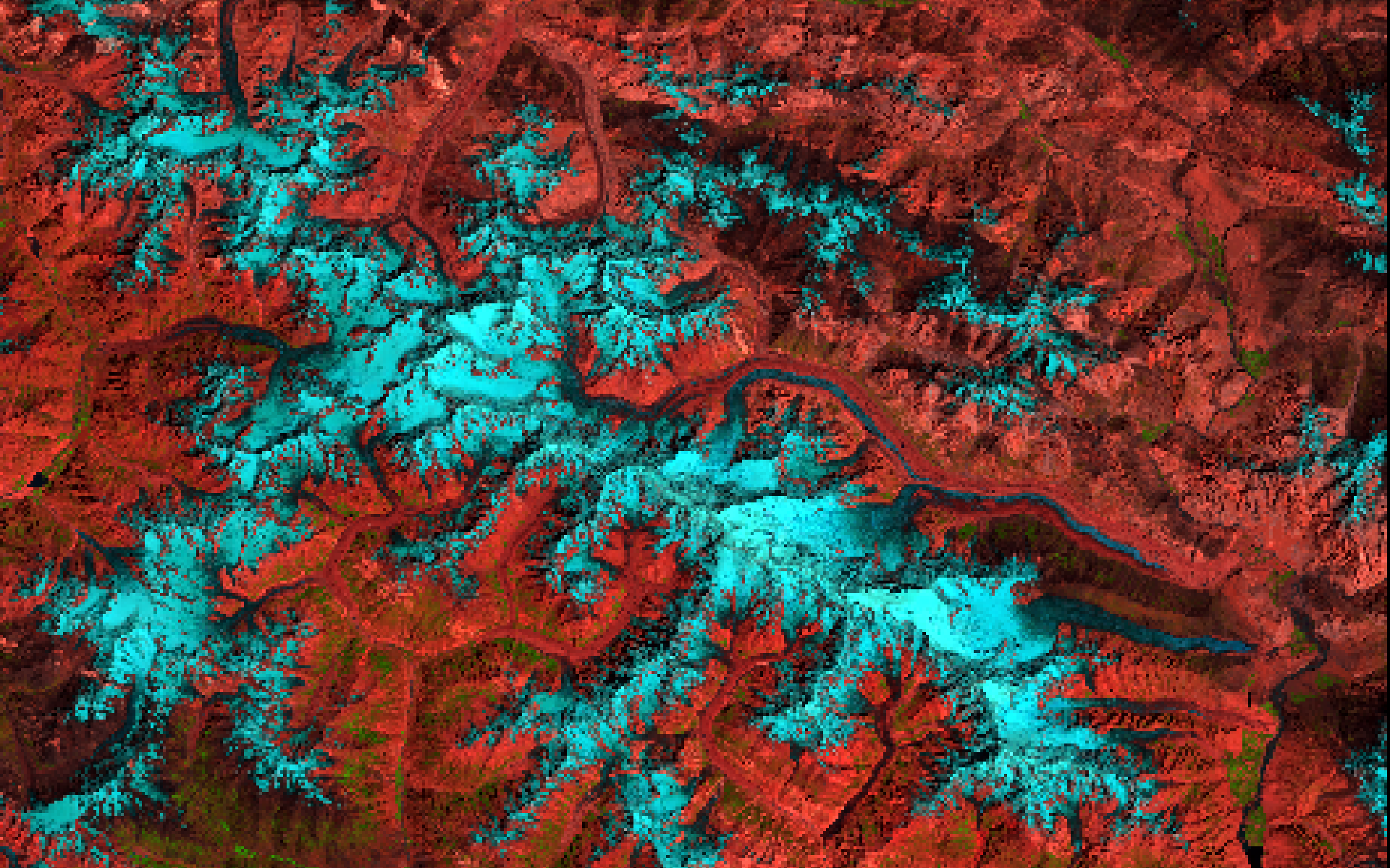
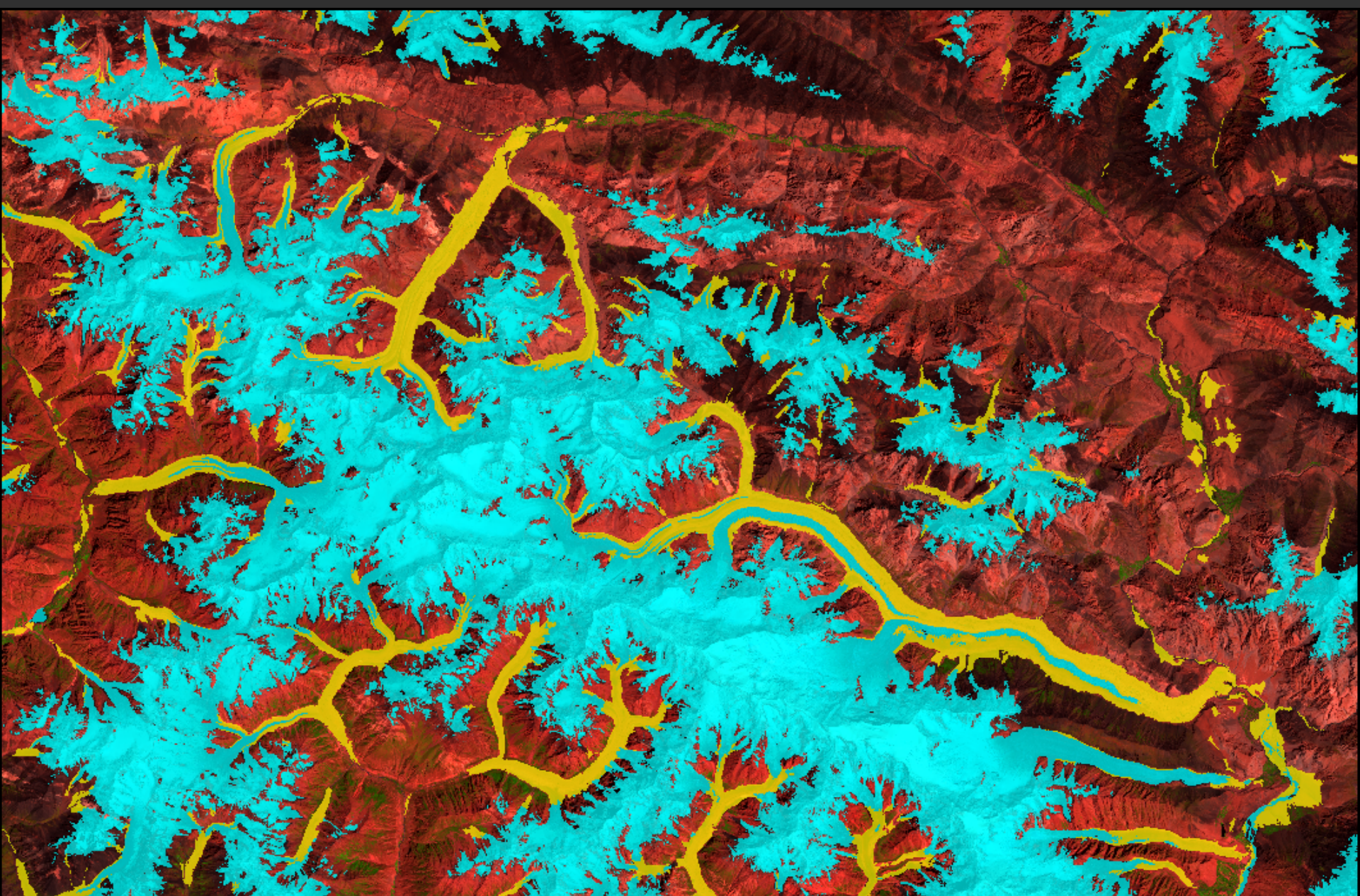
HeatMap



OBIA refinement



- classes
- glacier
 - clean ice glacier
 - debris covered ice
 - potential debris-covered glacier
- lakes
- shadows
- stable_slope
- vegetation



FINAL RESULTS

CONTACT

ARIOLUWA ARIBISALA

arioluwa.aribisala@stud.sbg.ac.at

SOFIA DELGADO

sofia.delgado-balaguera@stud.sbg.ac.at

SAAD AHMED JAMAL

saad.jamal@stud.sbg.ac.at

OSCAR NARVAEZ LUCES

oscar.narvaez-luces@stud.sbg.ac.at

SARAH POCH

sarah.poch@stud.sbg.ac.at

ZHIBEK SOLPIEVA

zhibek.solpieva@stud.sbg.ac.at



COPERNICUS MASTER
IN DIGITAL EARTH



PARIS
LODRON
UNIVERSITÄT
SALZBURG



THANK YOU FOR YOUR
ATTENTION!

REFERENCES:

ROBSON Benjamin and THOMAS Daniel , Mapping debris-covered glaciers using Convolutional Neural Networks and Object-Based Image Analysis
BRYNER, Jeanna (April, 2017) . Photographic Proof of Climate Change: Time-Lapse Images of Retreating Glaciers. Live Science. <https://www.livescience.com/58774-time-lapse-photos-show-retreating-glaciers.html>