Course : Image Processing for Earth Observation

Mid term examination

First Name :

Last Name :

SCIPER Number:

Course of study :

The mid-term is composed of 7 questions. **Please answer all the questions.** Motivate your answers and report your calculations where applicable.
The length of the answer boxes is <u>not</u> proportional to the expected length of the answer. You can use the back of the sheets to write if really you need more, but we prefer concise answers.

You can use all the material provided during the course.

Good luck!

Lausanne, 10.12.2021

Question 1 (2 points)

You are working in an NGO and have been contacted by concerned citizens about habitat loss of the critically endangered mountain gorillas in Rwanda. The country is not only home to the Volcanoes National Park, where the majority of mountain gorillas live, but also to a thriving human population, which performs increasingly expansive agricultural practices in the surroundings of the park.

In this context, you have been asked to estimate whether and by how much the mountain gorillas' habitat in the area has been affected. For long-term conservation purposes, the enquirers are particularly interested in changes in habitat area over the last three to four decades.

Sketch how you would design your processing pipeline, including data acquisition, design and deployment. You have a very limited budget.

Conditions:

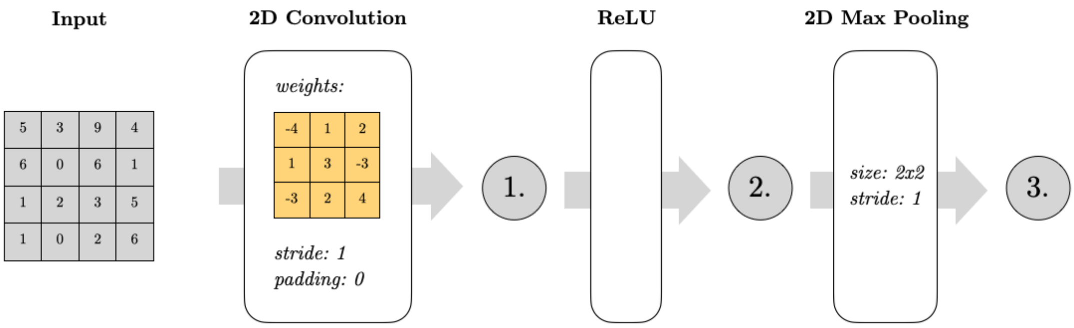
1. Topic: change detection, forested areas/deforestation
2. Time scale is three to four decades → Landsat (30m resolution is enough)
3. Small budget → free satellite data (Landsat); no expensive models (preferably no DL), no crowdsourcing to get labels

Possible solution:

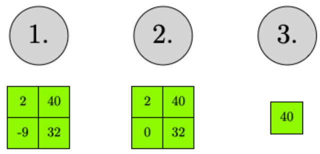
1. Download Landsat-5 and -8 imagery over the Volcanoes National Park area, with minimum cloud cover
2. Draw an initial set of coarse polygons of forested and non-forested areas for each timestep
3. Split the data into train/val/test sets (same spatially disjoint regions across all timesteps)
4. Extract manual features with given neighbourhood around each pixel: NDVI, BovW, histogram of colours, etc.
5. Train an initial model, such as a Random Forest, an SVM, etc. Can be one for each timestep (if sufficient number of initial polygons) or one general model.
6. Crossvalidate hyperparameters w.r.t. validation set (no. trees, tree depth)
7. Change detection: predict all pixels and calculate difference in forested areas. Sum deforested and newly forested changes across each timestep and provide true areas (no. pixels multiplied by pixel size)

Question 2 (1.5 points)

You are provided an image with pixel values and a CNN with three layers and indicated parameters:



Calculate and draw the outputs of each of the CNN's layers (1., 2. and 3. in the figure). Explicit your calculations.



Question 3 (1.5 points)

Which sensor would you use for which application? Complete the table below.

Task	Sensor	Why?
Monitor floods status during a storm	Sentinel-1 (SAR)	SAR can see through clouds
Monitoring parking usage of large supermarkets	Optical aerial imagery	Need for high resolution, optical bands are sufficient
Detect ripe tomatoes in fields for precision harvesting	RGB+NIR camera mounted on UAV	Need for very high resolution, NIR helps detect the condition of the tomatoes
Map aerosols over Northern America	Hyperspectral satellite (Sentinel 4)	Satellite for large coverage. Hyperspectral for detecting aerosols
Monitor deforestation	Landsat or Sentinel-2 RGB+NIR	Satellite for large coverage. RGB+NIR to compute NDVI

Question 4 (1 point)

The code below is supposed to train an AlexNet deep CNN on the ImageNet dataset in PyTorch:

```
import torch
from torch.utils.data import DataLoader
import torchvision

dataLoader = DataLoader(
      torchvision.datasets.ImageNet('/data/imageNet')
)
model = torchvision.models.AlexNet(num_classes=1000)
optim = torch.optim.SGD(model.parameters(), lr=1e-3)
criterion = torch.nn.CrossEntropyLoss()

for (data, labels) in dataLoader:
   pred = model(data.cuda())
   labels_pred = pred.argmax(1)
   loss = criterion(labels_pred, labels)
   loss.backward()
   optim.step()

   # print overall accuracy
   print(torch.mean((labels_pred == labels).float()).item())
```

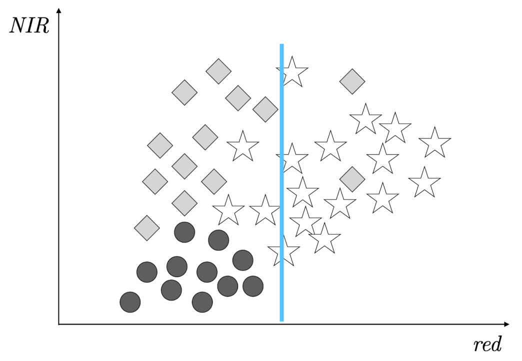
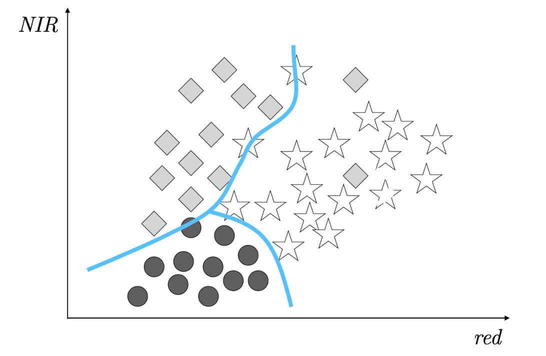
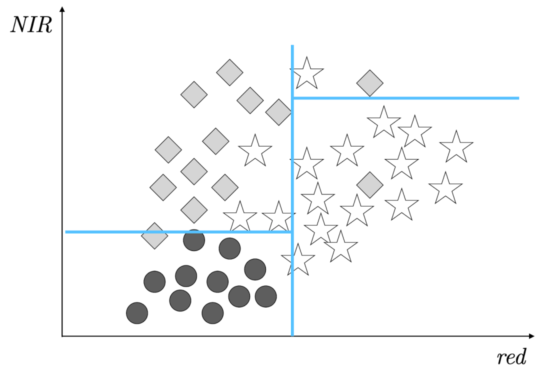
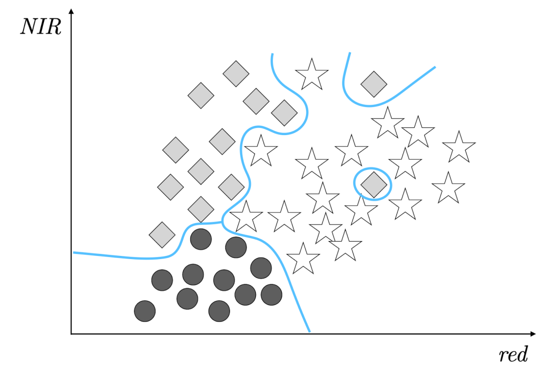
Unfortunately, three errors have made it into this code snippet. Please identify two of those errors and explain what problem they would cause.

You can assume the ImageNet dataset to be properly downloaded and configured under `/data/imageNet`.

1. Only the data is put on the GPU (data.cuda()); the model and labels are not. This causes code execution to abort.
2. There is no optim.zero_grad() statement, causing the gradients to accumulate across all batches → code runs, but exploding gradients and bad learning signal.
3. The input to the loss function should be pred, not labels_pred → execution aborts due to shape mismatch (and device error; see 1.).

Question 5 (1.5 points)

Given a set of data points and their intensity values in red and near-infrared as follows, draw the decision boundaries for the indicated classifiers:

	
Decision tree of depth 1	Random forest with 500 trees
	
Decision tree of depth 2	k-Nearest Neighbours with k = 1

Can you identify the types of land cover classes if we assume these pixel values come from a satellite image? Assume both axes are shown from 0 to the maximal radiometric resolution.

- Diamonds: vegetation
- Circles: water
- Stars: soils, other materials

Question 6 (2 points)

You are given 100 aerial images over a residential area like the one shown below:

They are 0.5m resolution, orthorectified images with four bands: red, green, blue and near-infrared.

You have been asked to map all building footprints. Your predecessor on the project already started labelling a few, but could only annotate eight (8) such images with polygons due to time restrictions.

Design a plan to get from the status quo to the required output, detailing the intermediate steps as much as possible. Also indicate what performance metrics you plan on using.

Possible solutions:

Without DL **(not full points)**:

1. Assign eight labelled images to train/test sets
2. Run SLIC on all the images (optimising for region separability w.r.t. the given polygons in the training set)
3. Calculate features: (NDVI), BoVW, histogram of colours, etc.
4. Train classifier (Random Forest, etc.) on training set, performing k-fold crossvalidation on the no. trees and tree depth
5. Train final model; predict on test set to get performance score (OA; precision, recall, F1 score for footprints)
6. Predict on all images and save footprint masks

With DL + auxiliary data:

1. Download additional building footprints from OpenStreetMap.org
2. Assign images to train/val/test sets (e.g., 60/10/30 percent), separately for the 92 unannotated images (set A) and the eight labelled ones (set B)
3. Split images into tiles of reasonable size (e.g., 800x600)

4. Design a semantic segmentation CNN, e.g., U-Net, Hypercolumn; pre-trained on ImageNet where possible
5. Train the model with SGD and BCE loss on the tiles and the OSM data
6. Optimise the hyperparameters with validation set A
7. Fine-tune the model on the fully annotated images (no layer/parameter replacement needed), with a slightly smaller learning rate
8. Optimise the hyperparameters with validation set B
9. Predict on test set to get performance scores (OA; precision, recall, F1 score for footprints)
10. Predict on all images and save footprint masks

Question 7 (1.5 points)

TearWallsApart Inc. is a company that builds thermal cameras, which are used to automatically detect faults in façades. You are given the task to build a classifier that detects if a wall is about to crack. Missing many would be a catastrophe for the people living in the house.

With that in mind, you try various classifiers and come up with the following confusion matrices and overall accuracy scores. By using only this information, which classifier do you choose? Motivate your answer in your own words.

Classifier 1		PREDICTED			
		Good as new	Under stress	About to crack	Cracks visible
TRUE	Good as new	5000	0	0	100
	Under stress	100	5000	0	0
	About to crack	2000	500	700	300
	Cracks visible	0	100	0	1200

Overall accuracy: 79.33%

Classifier 2		PREDICTED			
		Good as new	Under stress	About to crack	Cracks visible
TRUE	Good as new	5000	0	0	100
	Under stress	300	3700	1100	0
	About to crack	100	600	2200	600
	Cracks visible	0	100	600	600

Overall accuracy: 76.67 %

Classifier 3		PREDICTED			
		Good as new	Under stress	About to crack	Cracks visible
TRUE	Good as new	4700	300	0	100
	Under stress	300	4400	400	0
	About to crack	200	1200	2100	0
	Cracks visible	0	500	0	800

Overall accuracy: 80.00 %

Since no cracking walls should not be missed, producer accuracy is more important that user accuracy. In other words, it is better to overestimate the damage than underestimate the damage.

Classifier 1 classifies most “about to crack” walls as “good as new”.
Classifier 3 has more “about to crack” walls classified as “good as new” or “under stress” that classifier 2.

The best classifier is thus **classifier 2**.