Internship Report

ING2

Leveraging deep learning and remote sensing to predict ecosystem types in the NiN framework

Image generated by stable diffusion with the following prompt *'two panels linocut featuring a satellite hovering above a boreal forest on the left, and a neural network on the right'.*

Matteo CRESPIN-Jouan

September 2024

Host institution

Geo-Ecology Research Group (GEco), at Oslo's Natural History Museum (NHM)

Sars' gate 1, 0562 Oslo, Norway

Tutor at the host institution

Anders Bryn, Professor

Referent teacher at the sending institution

Alexandre Hippert-Ferrer

Rapporteur principal

<Prénom NOM>

Internship from April 29th 2024 to July 12th 2024

Number of pages : 29

Version : Final

Acknowledgements

I spent 11 lovely weeks at GEco that were truly stimulating, both intellectually and socially. I owe a deep debt of gratitude to Anders, who, after being an exceptional professor during the semester leading up to this internship, not only offered me the chance to work with this data but also invited me to join a field week to witness firsthand how it is gathered. My sincere thanks also go to Adam, Peter, and Ingrid, who graciously answered my many questions and showed immense patience in the face of my nearly complete lack of knowledge in ecology, botany and Norwegian.

Résumé

Ce rapport présente les résultats d'un stage effectué au sein du Geo-Ecology Research Group (GEco) du Muséum d'Histoire Naturelle d'Oslo. Le projet a porté sur l'application de techniques d'apprentissage profond pour classifier les écosystèmes norvégiens en se basant sur les données du système de classification Natur i Norge (NiN). Différentes sources de données ont été utilisées, notamment des images aériennes de drones, des photos prises au sol et des données satellitaires Sentinel, afin de prédire les types d'écosystèmes et des gradients environnementaux clés, tels que la richesse en calcaire.

L'étude a exploré différentes approches, notamment les réseaux neuronaux convolutifs (CNN) et les perceptrons multicouches (MLP), en mettant l'accent sur l'exploitation des informations spectrales plutôt que des caractéristiques spatiales. Les résultats ont mis en évidence les défis liés au travail avec des données limitées et incohérentes, en particulier dans le contexte de classifications très détaillées comme NiN. Bien que les modèles aient montré un certain succès, notamment avec l'utilisation de données hyperspectrales, les résultats ont été limités par la qualité et la cohérence des labels disponibles.

Mots-clefs : Apprentissage profond, Classification de la nature, Télédétection, Sentinel-2, Cartographie de la végétation, Cadre NiN, Réseaux de neurones convolutifs (CNN), Occupation et couverture des sols (LULC), Gradients environnementaux

Abstract

This report presents the results of an internship conducted at the Geo-Ecology Research Group (GEco) of the Natural History Museum of Oslo. The project focused on applying deep learning techniques to classify Norwegian ecosystems based on data from the Natur i Norge (NiN) classification system. Various data sources were used, including aerial drone images, ground-level photos, and Sentinel satellite data, to predict ecosystem types and key environmental gradients such as lime richness.

The study explored different approaches, including convolutional neural networks (CNNs) and multi-layer perceptrons (MLPs), with a focus on leveraging spectral information over spatial features. The results highlighted the challenges of working with limited and inconsistent data, particularly in the context of highly detailed classifications like NiN. Although the models showed some success, particularly when using hyperspectral data, the results were limited by the quality and consistency of the available labels.

Keywords : Deep learning, Nature classification, Remote sensing, Sentinel-2, Vegetation mapping, NiN framework, Convolutional neural networks (CNN), Land Use and Land Cover (LULC), Environmental gradients

Table of contents

Introduction

About the Data, the labels, and the distribution of the labels in the datasets

I. The labels : what is Natur i Norge (NiN)
II. The distribution of the labels in the dataset
    a) Where does the data come from
    b) Consensus points
    c) A bigger but less reliable source of labels : polygons from single-mappers
III. The data
    a) Photos taken at the ground level
    b) Drone images
    c) Aerial ortho-images
    d) Digital elevation model and canopy height model
    e) Lidar data
    f) Sentinel data

CNNs and vision transformers to leverage shape and texture features

I. Pre-training from scratch vs fine-tuning
II. Transformers and CNNs
III. Results from across different datasets : ground pictures, drone images and ortho-images
    a) Some general technical details
    b) Leveraging the ground images (with the consensus points from Landsvik)
    c) Leveraging the drone images (with Ander's polygons from Landsvik)
    d) Moving on to the ortho-images, and shrinking the image sizes.

A more successful endeavour : a mere multilayer perceptron on hyper-spectral satellite images

I. Class prediction with a MLP and sentinel 2 data
    a) The pros of sentinel 2 data
    b) What is a MLP ?
    c) Building the dataset, and avoiding data leak between the train and validation set
    d) Results & comments
II. A failed attempt to leverage Sentinel 1
III. Predicting lime richness

Concluding remarks

INTRODUCTION

To most people, Oslo's *Naturhistorisk museum* is merely a museum nested in the botanical garden of the Tøyen neighbourhood showcasing various popular science exhibition about the history of earth and the many forms of life inhabited by it. More inconspicuously perhaps, it is in fact, just like the *Muséum d'Histoire Naturelle* in Paris, a research institution : behind its art deco façade lies indeed a vast array of biology and geology research laboratories linked to the University of Oslo. Some of them deal with evolution (one subfield of biology), like the *Evolution and Paleobiology* (EPA) group, or the *Sex and Evolution Research Group* (SERG), others with genomics (EDGE group, CEG group), and one in particular, the Geo-Ecology Research Group (GEco) deals with one often less discussed subfield of biology : ecology.

What is ecology ?

In academic parlance, ecology refers specifically to the study the interactions between individuals and with their environment. Ecology itself is strongly structured in subfields, like population ecology (how the individuals within one species interact with each other and the environment, *eg* the modelling of population dynamics of foxes), community ecology (how *species* interact with each other,), or ecosystem ecology. The latter aims at modelling ecosystems as a whole, often through the lenses of the fluxes of matter and energy — namely, the fluxes of carbon — from the air to the living biomass and the soil. In that perspective, an ecosystem is merely *a certain way* for the carbon to cycle, as shown in the adjacent diagram, and the *structure* of an ecosystem pertains to the size of the different fluxes, as shown in the diagrams below. Depending on the species and climate involved, the structure of the ecosystem (the weights of the various arrows) can vary significantly. For example, the metabolism of the ecosystem is much faster in tropical forests because high amounts of solar radiation allow for greater synthesis of carbohydrates. This enables plants to build more biomass and provides decomposers with more energy to break down litter (and return carbon to the atmosphere).

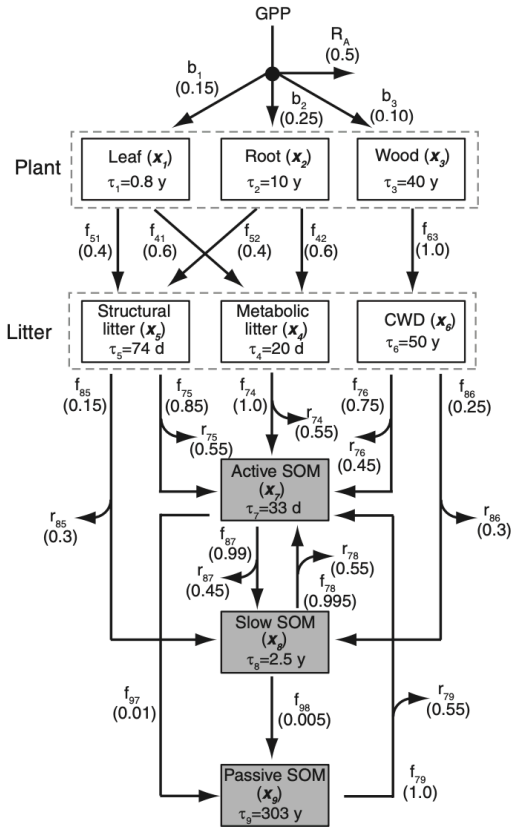


Figure 1 : General representation of a terrestrial ecosystem, from G. Bonan, *Ecological climatology*.

In the diagram below, GPP (gross primary production) is the total amount of carbon that is absorbed from the atmosphere. Some of this carbon is immediately respired to produce energy (R_a, autotrophic respiration), and the rest is used to build the tissues that the plants are made out of : this biomass buildup is called NPP, net primary production. But not all the built biomass actually builds up : as annual plants die and deciduous perennial shed their leaves, some of this living organic matter is transferred to the soil, where some of it is, in turn, respired by the decomposers (this is R_h, heterotrophic respiration). NEP, or net ecosystem production, is the amount of carbon that actually accumulates through these biotic processes, ie GPP - R_h - R_a.

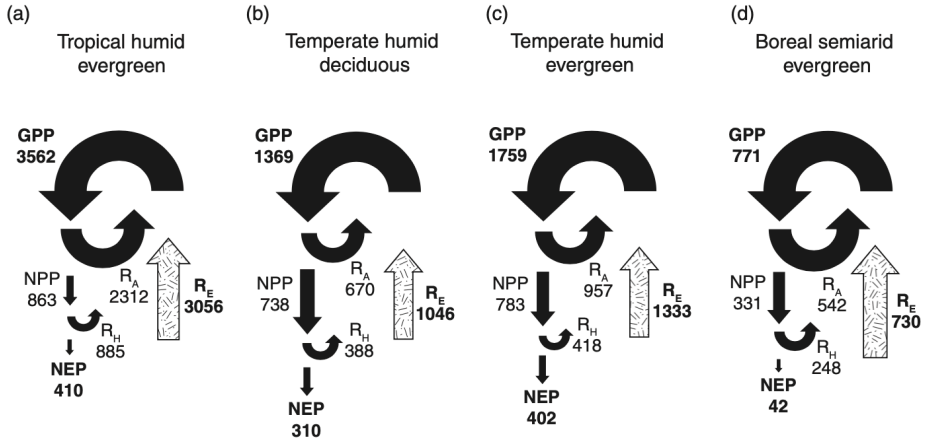


Figure 2 : Annual carbon fluxes (g C m-2 yr-1) for different ecosystem types G. Bonan, *Ecological climatology*.

Thus, one readily understands that being able to characterise ecosystem types is crucial, merely to know where carbon is being trapped or released and in what amount. For instance, in wetlands, the soil organic matter is surrounded by water, so the absence of oxygen makes it impossible for most decomposer to respire the carbohydrate chains — and oxidise carbon into deadly, planet wrecking CO2. In mountainous areas, snow cover lasting well into the spring can dramatically impede the growing season, as plants cannot photosynthesise CO2 into carbohydrates before the sun shines on their leaves, leading to very slow metabolisms — and very different species distribution.

What is GEco about ?

The research group where my internship took place, GEco, short for Geo-Ecology Research Group, precisely deals with these aspects of ecology. It is a small structure with 10 employees and 4 master students spread out across three floors at the botanical museum that carries research in fields like distribution modelling (eg understanding the position of tree-lines in the mountains), population biology, and vegetation mapping. One of their flagship long term project is the *Natur I Norge* (Nature in Norway, NiN) project, that has been running since 2005 under the guidance of *Artsdatabanken*, a.k.a. the Norwegian Biodiversity Information Center, that aims at developing a framework to classify and map ecosystems in Norway. It is from this classification — that I will expatiate on in greater detail in part one — that I have been training deep learning models during my internship.

Not only do the GEco researchers devise a classification system for nature types in Norway, but they also take part in mapping themselves to probe it against the practicalities of the field. But here's the catch : when different mappers, even very experimented ecologists, go on the field, they very often comme back with different classes. This is probably not a surprise when one knows the vast number of classes that NiN contains (close to 300 at the lowest level of the classification, a far cry from Corine Land Cover !), but the extent of the inconsistencies is a matter of concern that the GEco group strives to address. The figure below, from a 2017 paper, shows the level of the inconsistencies : only the green surface is where the three mappers came up with the same label. The inconsistencies often stem from displaced boundaries (the mappers do not delineate nature

types similarly when they draw polygons), but also from entirely different classes : unsurprisingly, at the lowest level of the classification, and to a lesser extent but devastatingly still, at the highest level (*Ullerul & al, 2017*).

What would I be doing at GEco ?

To try and address this issue and even out their classification decisions, some researchers of the GEco group took time to reach an agreement about the nature types in a few hundred of points across two locations in Norway. When I was offered this internship, they thus proposed me to leverage this consistent dataset with deep learning techniques, with the idea that it may perhaps yield interesting results generalisable to other locations.

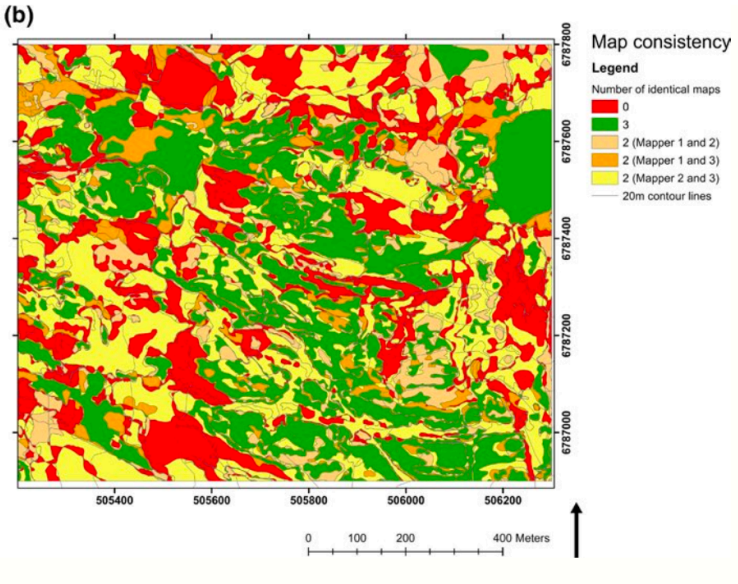


Figure 3 : Inconsistencies between three mappers at the lowest NiN classification level, from *Ullerud & al, 2017*

For each of the consensus points, they had one picture taken at the ground level, as well as three or four channels drone images, and a drone-borne Lidar scan of the scene. The idea would to explore different avenues to see if deep learning techniques could work to assign NiN classes to aerial images.

What has been done so far ? A brief review of the literature

The use of remote sensing for land use and land cover (LULC) mapping traces back to the onset of remote sensing, with the first nation-wide aerial images of the United States in the 1940s (*Marschner, F, 1950*), but it has gained traction in the past few years due to the rapid and successful development of deep learning techniques that took over other machine learning approaches (*A. Vali & all, 2020*) and enabled to leverage the vasts amount of data collected by earth observation satellites.

Because earth observation satellites have already been around for a while, the developments in this field followed pretty much the more general trends in deep learning for image analysis, merely with a few years delay, and thus boomed in the wake of CNNs gaining widespread attention after AlexNet won the ImageNet Large Scale Visual Recognition Challenge in 2012. This doesn't come as a surprise when one considers that LULC classification is merely the application of image analysis to earth images : the same network that is trained to segment cars in an image or tell whether it shows a cat or a dog can very well be trained to segment rivers or tell whether it's a desert or a rainforest.

Current techniques are overwhelmingly dominated by convolutional neural networks (CNNs) and their likes (ResNets, UNets). However RNNs (recurrent neural networks) are still used for specific applications like time series, and vision transformers —based on the *self-attention mechanism* that sparked the revolution in natural language processing — are on the rise (*S. Zhao & all,* 2023).

There are three main ways to handle the problem of land use and land cover (LULC) classification : the simplest one is **pixel level classification** : based on the information at the pixel level, one can assign such or such class to the pixel. But the advent of deep learning ushered in more sophisticated techniques. One can for instance train a model to label directly at the image level, extracting high level features compounded from across the entire image — does it show a boreal forest ? An airport ? A beach ? A field lying fallow ? This is

called **scene classification**, and can typically be done with a model like ResNet (*A. Thapa & all,* 2023). The problem is that if the image shows a beach, it probably also shows a body of water on one side and vegetation on the other, so one general label for the scene wouldn't be convenient for mapping purposes. A much more handy way to directly map vast stretches of land is **semantic segmentation** : here, the model receives one image, and just like in pixel level classification, assigns a class to every pixel. But this time, the class isn't based so much on the pixel itself as on the whole input, relying on long-distance dependencies across the image : it can hence tell whether a pixel representing water is to be labeled as a lake, a river or a pond based on broader context, even if they have exactly the same spectral signature.

However, these techniques have mostly been applied to LULC classification framework very different from that of NiN. One recent meta-analysis (*A. Thapa & all,* 2023) looked at the dataset used by papers carrying scene classification on aerial images, and it appears that these dataset contains a fairly small number of classes that are way broader than that of NiN. For instance, the Aerial Image Dataset (AID) contains 30 classes, with very general labels like residential, industrial, commercial or agricultural. The WHU-RS19 dataset contains a mere 19 classes, as broad as 'desert, farmland, footballfield, forest, industrial or meadow'. On the other hand, the nature types at the finest level of NiN make very subtle distinctions, even to the naturalist's eye, for instance, within the broad category *Åpen grunnlendt mark*, that encompasses all naturally open, soil-covered land where there is no basis for forest formation due to a thin soil layer, it distinguishes between 8 classes, such as *Åpen kalkfattig grunnlendt lavmark* (French ≈ *Terrain ouvert pauvre en calcaire avec des lichens sur sol peu profond*) and *Åpen kalkfattig grunnlendt lyngmark* (French ≈ *Terrain ouvert pauvre en calcaire avec de la bryère sur sol peu profond*), and these two categories are broken down along 4 levels of lime-richness, ranging from very acidic to very alkaline soils.

The closest land cover classification framework that boasts a significant body of deep learning LULC classification research is probably Copernicus's Corine Land Cover 2018 (CLC18), with its 45 thematic classes. But even here, where CLC18 distinguishes between 3 kinds of forests, NiN has more than 20. Where CLC has 5 wetland types, NiN has close to a hundred at the finest classification level. In that regard, NiN lies in a league of its own, and the works that can be cited used way coarser land use classification frameworks. The vast majority used a U-Net architecture for semantic segmentation : *D. B. Demir & all*, 2023, on level 2 classes oc CLC (namely, 15 different classes), or *Vasilis Pollato & all*, 2020 on 32 of the 45 level 3 classes. Both papers pointed out to the quality and size of the dataset as the limiting factor for the performance of their models.

It is not, however, in that direction that I have decided to go, in large part because of the nature of my dataset, as I will expose it in part I. I most of the experiments that I ran, I opted either for the scene classification approach, or for simple pixel level classification with fully connected neural network. This is far from the state of the art, but the data that I was dealing with was also very different from the one semantic segmentation is generally performed on.

The present report is structured in 3 parts. First, I expatiate on different aspects of the dataset (what is Nature I Norge, labels, data, distribution of labels). Second, I will expose the different methods that I used to try and leverage shape and texture data. Third, I will detail the experiments that I ran using the rich spectral data of sentinel 2. In the conclusion, I give some suggestions of directions to continue this work.

ABOUT THE DATA, THE LABELS, AND THE DISTRIBUTION OF THE LABELS IN THE DATASETS

I. The labels : what is *Natur i Norge* (NiN)

Natur i Norge is a classification of nature types in Norway. It is a three level hierarchical classification that covers all existing ecosystems in Norway.

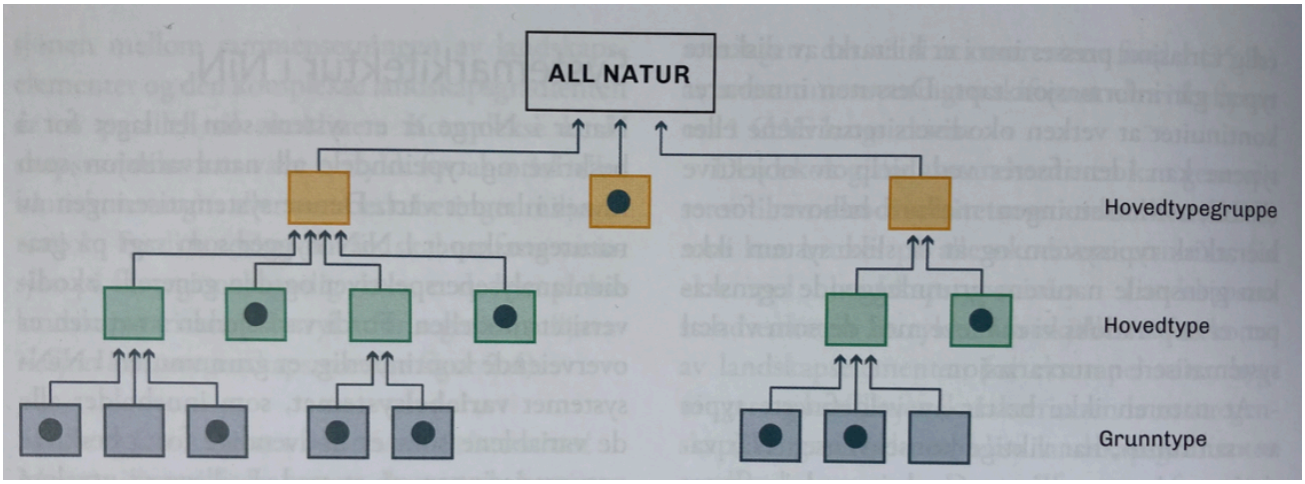


Figure 4 : Schema of the hierarchical structure of Nature I Norge, from the Nature I Norge book.

First level of the classification : *Hovedtypegrupper* ('main type group')

At the highest level of the classification, there are *hovedtypegrupper*, literally 'groups of main types'. For land, I only had to deal with *våtmark* — wetlands — and *fastmark* — drylands, or solid ground.

As depicted in the diagram below, the *hovedtypegupper* are subdivided into *hovedtyper* (main types), and the main types are divided into *grunntyper*, ground types.

There have been many iterations of the NiN classification, and although I used data mapped under NiN 2.3, the educational material that I had access to was describing NiN 3.

Second level of the classification : *hovedtyper* ('main type')

Either way, the *hovedtypegrupper* (level 1) are subdivided into *hovedtyper* (level 2) based on « process categories » that represent the most important ecological processes at play in shaping the nature type in a given area. In the *'dryland' hovedtypegruppe* (level 1) for instance, one of the process category is called *miljøstress*, and refers to non-human environmental factors that hinder the vegetation development in a permanent way (for instance, a long lasting snow cover), in such a way that it is this, rather than the competition among species, that is primarily responsible for the observed distribution. Another process category is *aktiv destabiliserende forstyrrelse* (active destabilising disruption), and rather refers to a non-human punctual disruption that regularly resets the ecological succession. One example (which then leads to one corresponding *hovedtype* — level 2) is flooding. If an area is flooded every few years, all it can host is early-successional pioneer species that are very good at photosynthesising at full throttle to colonise the recently flooded environment, but that are also very bad at competition and would be outcompeted by late-successional species if the latter had time to grow. If the trees can never settle and deprive the pioneer species of access to light, then the flooding is the main process behind

the observed species distribution, hence the *hovedtype* (level 2) « åpen flomfastmark », literally « open flooding dryland ».

Third (and last) level of the classification : *grunntyper* ('groundtypes')

The main ecological process determines the maintype (level 2, in the flooding example below, TE03), but there are many local factors that contribute to a given distribution of species. Very often, the lime content in the soil looms very large in the species distribution. The desiccation risk too : even if it is a wetland, it may dry out sometimes, and whereas some species do not mind a few weeks with less water, other would die out. Another one is the size of the soil particles : is it silt ? Is it sand ? Is it gravel ? Is it big blocs ?

These factors are not necesarlily the proximate cause for the species distribution. For instance, lime content certainly plays a role, but perhaps it is mostly through the fact that it co-varies with acidity, which in turn, determines the kind of nutrients that are dissolved in the soil solution. Since acidity and lime richness and calcium content covary very much, they are lumped together under one 'local complex gradient', LKM, called *kalkinhold* (lime content). For each *hovedtype*, a handful of variables are selected, forming a hypervolume (the *environmental space*) within which the *grunntyper* (level 3) are delineated. It can be a simple 2-D plane, like on the example below, where on the y axis, the LKM is *kalkinnhold* (lime richness), ranging from medium to high, and on the x axis, it is the dominant grain size, that varies across clay, silt, sand, gravel, stone, blocs and big blocs.

<table>
<tr><td rowspan="5">Lite endret fastmark</td><td colspan="2">TE03 Åpen flomfastmark</td><td colspan="8"></td><td>KE = 6
GT = 6</td></tr>
<tr><td rowspan="3">KA – Kalkinnhold
Svært kalkrik
Svakt intermediær</td><td>gh</td><td rowspan="2">TE03-01 Intermediær til sterkt kalkrik åpen leire-flomfastmark</td><td rowspan="2">TE03-02 Intermediær til sterkt kalkrik åpen silt-flomfastmark</td><td colspan="2" rowspan="2">TE03-03 Intermediær til sterkt kalkrik åpen sand- og grus-flomfastmark</td><td colspan="2">TE03-05 Sterkt kalkrik åpen stein- og blokk-flomfastmark</td><td rowspan="2">TE03-06 Åpen storblokket flomfastmark</td></tr>
<tr><td>def</td><td colspan="2">TE03-04 Intermediær til litt kalkrik åpen stein- og blokk-flomfastmark</td></tr>
<tr><td colspan="2"></td><td>A</td><td>B</td><td>C</td><td>D</td><td>E</td><td>F</td><td>G</td></tr>
<tr><td colspan="2"></td><td>Leire</td><td>Silt</td><td>Sand</td><td>Grus</td><td>Stein</td><td>Blokk</td><td>Stor blokk</td></tr>
<tr><td colspan="2"></td><td colspan="8">DK – Dominerende kornstørrelse</td></tr>
</table>

Figure 5 : Environmental space in which 6 ground types (level 3) are delineated for the main type (level 2) 'open dry land liable to flooding', from *A. Bryn & all, 2023*

Below, the ground types (level 3) from the 'forest on dryland main type' (level 2) are delineated within a 3D volume made out of the following local complex gradients : *i.* the lime content, *ii.* the desiccation risk *iii.* the average water saturation level. The z axis, where VM is the water saturation is representing by showing two 2D planes (as a way to unwind the 3D volume).

Lite endret fastmark	TB01 Fastmarksskogsmark					KE = 18 / GT = 18
	KA – Kalkinnhold: Ekstremt kalkrik ↑ Temmelig kalkfattig	ghi	TB01-03 Frisk kalkskog	TB01-06 Kalk-bærlyngskog	TB01-09 Kalk-lyngskog	TB01-12 Kalk-lavskog
		def	TB01-02 Lågurtskog	TB01-05 Bærlyng-lågurtskog	TB01-08 Lyngskog	TB01-11 Lav-lågurtskog
		bc	TB01-01 Blåbærskog	TB01-04 Bærlyngskog	TB01-07 Lyngskog	TB01-10 Lavskog
	VM_0a		ab	cd	ef	gh
			Frisk ----------> Ekstremt tørkeutsatt			
			UF – Uttørkingsfare			
	KA – Kalkinnhold: Ekstremt kalkrik ↑ Temmelig kalkfattig	ghi	TB01-15 Høgstaudeskog		TB01-18 Kalkrik lyngfuktskog	
		def	TB01-14 Storbregneskog		TB01-17 Intermediær lyngfuktskog	
		bc	TB01-13 Blåbærfuktskog		TB01-16 Lyngfuktskog	
	VM_bc		ab		cd	ef
			Frisk ----------> Ekstremt tørkeutsatt			
			UF – Uttørkingsfare			

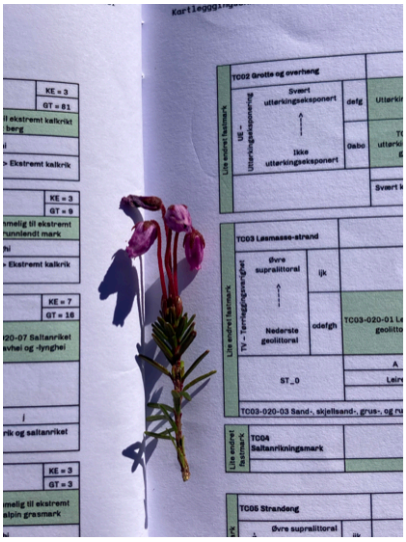
Figure 6 : Environmental space in which 18 ground types (level 3) are delineated for the maintype (level 2) 'dryland forests', from *A. Bryn & all, 2023*

But how are the classes mapped on the field ?

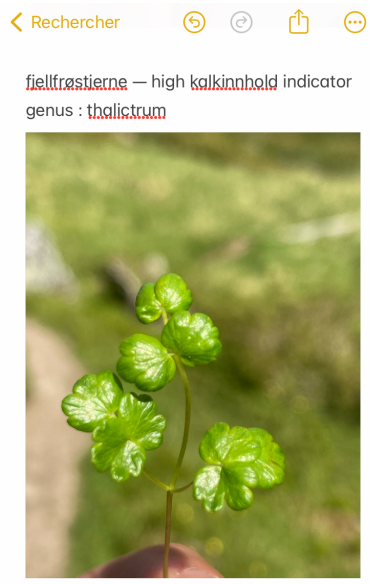
This is an important question to get an idea of what kind of data the models could need, and for that reason I spent 5 days in the field learning how the nature types are actually mapped.

Since it would be extremely tedious to measure everywhere the lime richness, and knowing whether the soil is generally saturated would require to come and check often through the year to calculate an average, these *local complex gradients* along which the ground types are delineated in the environmental space are not measured directly. Rather, ecologists use the observed species distribution as *proxies*, to try and figure out where in the environmental space an area lies.

Experimented mappers are ecologist that have a good understanding of the landscape dynamics, and can rapidly determine the main type (level 2) by figuring out what is the most shaping ecological process at play, but figuring out the *ground type* requires a careful inspection of the species on the ground, for instance to try and spot some plants that are strong indicator of high -or poor- lime content.



Picture from the field



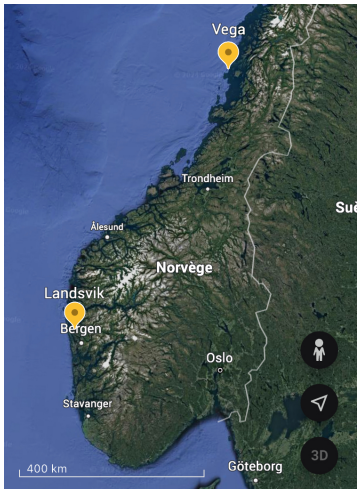
Example of lime richness indicator — screenshot from my herbarium

Just because aerial images are orders of magnitude away from being able to capture such tiny plants, it doesn't mean that other cues of lime richness cannot be discovered in the data : it may we be that other proxies lie hidden in the spectral signature of the leaves, and it's exactly the reason why deep learning models are so powerful. However this clearly suggests that ground images could be useful at some point in the process.

II. The distribution of the labels in the dataset

a) Where does the data come from

All the data that I used to train the model came from the same two localities : one is Landsvik and one is Vega. In Vega, it is split in three different localities across the island.



Location of Landsvik and Vega in Norway.

b) <u>Consensus points</u>

In the locality of Landsvik, in western Norway, I had 286 consensus points, for which the different mappers had taken time to come to an agreement.

These consensus points were distributed spatially and class-wise as shown in the figures below. One big issue for training was that the dataset was extremely imbalanced : only four classes occurred more than 30 times : T31-C-2 (lime-poor boreal heathland), T4-C-5 (berry heath forest), V1-C-5 (lime poor moor edges), and T1-C-2 (drought-prone, lime-poor rocks and cliffs). At the main type level (level 2 in the hierarchy), there was a bit less class diversity, as along the 53 T4-C-5, there was also 7 T4-C-2 (sparse herb-poor forest), and 9 T4-C-1 (blueberry forest), all of them belonging to the 'T4' forest category, referring to the forests on a non wet land.

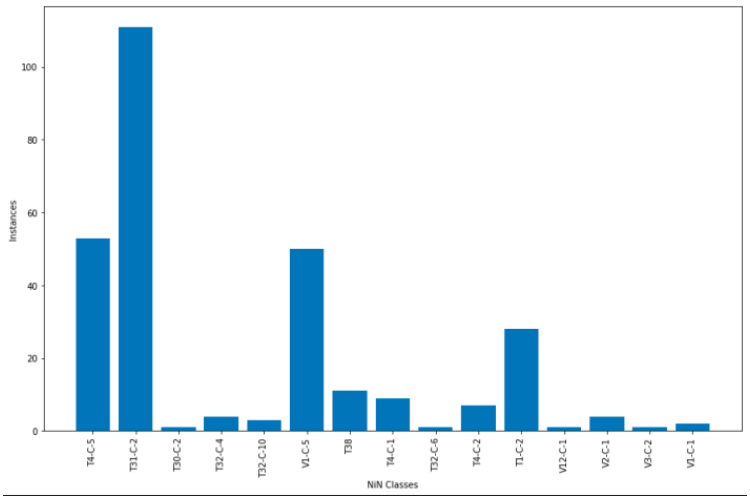


Figure 7 : Class-wise distribution of the consensus points in Landsvik

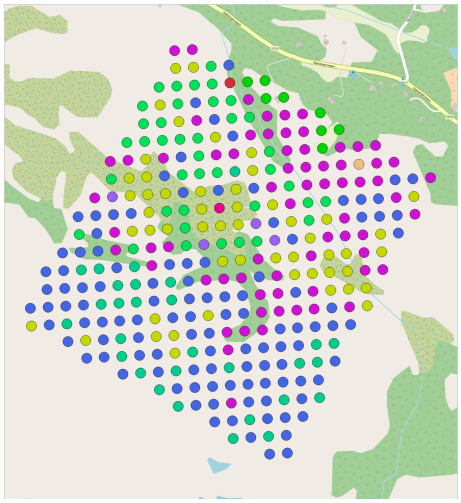


Figure 8 : Spatial distribution of the consensus points in Landsvik (screenshot from qgis)

Also, there should have been another set of consensus points from the other locality, but it had not been digitalised and one of the sheet had been lost. I didn't do it myself because it rapidly turned out there was way too few data anyways with the consensus points, and that I should use the polygons from single mappers instead.

c) A bigger but less reliable source of labels : polygons from single-mappers

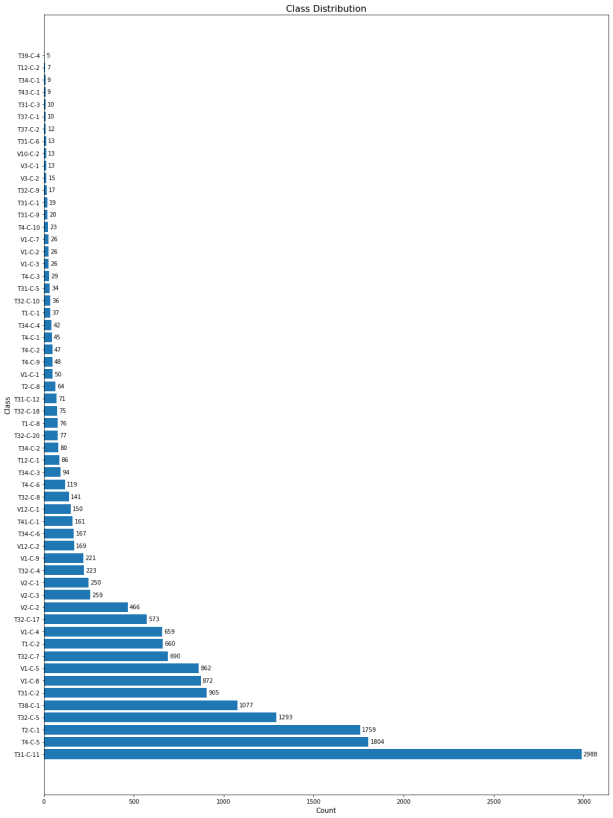


Figure 9 : Number of sentinel 2 pixels falling within the polygons of each labels as per Anders' mapping in Landsvik and Vega

I had also access to shapefiles that contained the labels of polygons for both Landsvik and Vega. This is a much richer data, because it gives a label for a surface, and not only for a point. So all the extent of one aerial image that lies within that polygon can be assigned the label. Where the consensus points allowed me to label only the Sentinel 2 pixel at the same location, each small polygon allowed me to label tens or hundreds of sentinel 2 pixel. On the screenshot above, the extent is about 650m per 750m. It also hosts much more classes, as shown on the barplot below that presents the number of sentinel 2 pixels that fall within the polygons mapped by Anders across both Vega and Landsvik. Once again, the distribution is utterly imbalanced, ranging from 5 occurrences to 2988.

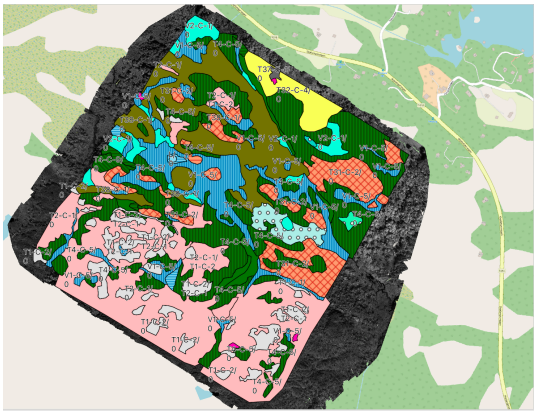


Figure 10 : Polygons from Anders in Vega (screenshot from Qgis)

The main issue with that approach is that these maps vary quite a lot from one mapper to another. Below for instance is Anders mapping of Landsvik (left) intersected with that of Rune : the polygons are shown only when the classes where the same at the *gruntyper* (lowest level, right), and at the *hovedtyper* (second to last, center). Unfortunately, the differences between the two mappers, experimented though they are, are so wide that the last two shapefiles look like lacework.

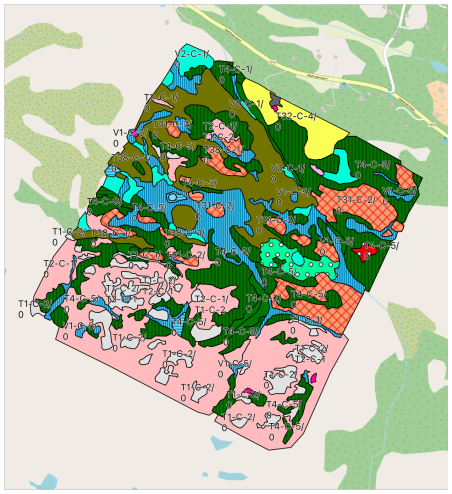


Figure 11 : Original Ander's Mapping

Figure 12 : Anders' mapping Intersected with Rune's at the hovedtyper level

Figure 13 : Anders' mapping Intersected with Rune's at the gruntyper level

III. The data

a) Photos taken at the ground level

For each of the 286 consensus points in Landsvik, one picture had been taken at the ground level with a camera. These images give cues that remote sensed data probably can't offer, especially when the species in the understory are hidden by the canopy : one the face of it, there's hardly any other way to determine the lime richness since it is denoted (at least to the mapper) by some small plants lying a few centimetres above the soil.

b) Drone images

For Landsvik, I had also access to drone images taken from above the canopy with a very high ground resolution (5.5 cm), and across green, blue, red, red-edge, near infrared, as well as one panchromatic channel. However I didn't have access to these images for Vega, so the dataset was fairly small.

Figure 14 : Drone image from Vega, screenshot from Qgis

c) Aerial ortho-images

For Landsvik and Vega, i had access to ortho-images with a 12.5 cm ground resolution. However, the data was a CIR image (coloured infrared, namely red, green and near infrared) for Landsvik, and an RGB one for Vega, so I couldn't use it across both locations because it would have been liable to the *clever hans effect* (cf Infra)

d) Digital elevation model and canopy height model

I was provided with digital elevation models and canopy height models at 0.1 m resolution. On the face of it, they give useful cues for the classes, for instance because the desiccation risk (which is a gradient used to divided the ecological space, and thus discriminate between classes at the *grundtyper* level) is not the same on a heap and on a hollow.

Another example is that the convexity or concavity of the terrain can loom very large on the length of the growing season. In mountainous areas the time slot where the temperature are high enough for the biological processes to occur is so small that a variation of 3 weeks of the snow cover can entail entirely different species distribution. Thus, there can be found, just 20 meters appart, some nature types falling within the *hovedtype* 'snow cover' (TC08), meaning that the duration of the snow cover is the main ecological process determining

species distribution, whereas on the convex area nearby, the vegetation grows unhindered into an alpine grassland (TA04).

However useful I think it could be, I didn't use this data because I headed down to other directions, and because my computer struggled very badly with the size of the files at hand. But I do think it is a way to be explored, because the infirmation it carries seems hands down as useful as the bands for the red edge area.

e) Lidar data

I was provided with drone-borne lidar scans of Landsvik, but here again I did not use it because Cloud Compare would crash the moment I tried to open the file, and also because I explored other leads of inquiry — a deception since I had done the research internship in the fall on Super Point Transformer, a deep learning model to handle point clouds.

f) Sentinel data

Cloudless late spring shots of sentinel 2 and radar acquisitions from sentinel 1 turned out to be the data that I used the most because it was in fact the only data available across both Vega *and* Landsvik (since the drone images from Vega wasn't available, and the ortho-images had different band). It is also because my experiments suggested that rich spectral information gave more useful cues than texture and shape one.

For Sentinel 2, I used all bands, except the two 60 m ones because they were coarser and meant for atmospheric processes. I used VH and VV bands from sentinel 1, but didn't understand how to access to the HH and HV ones.

Sentinel-2 Bands	Central Wavelength (µm)	Resolution (m)
Band 1 - Coastal aerosol	0.443	60
Band 2 - Blue	0.490	10
Band 3 - Green	0.560	10
Band 4 - Red	0.665	10
Band 5 - Vegetation Red Edge	0.705	20
Band 6 - Vegetation Red Edge	0.740	20
Band 7 - Vegetation Red Edge	0.783	20
Band 8 - NIR	0.842	10
Band 8A - Vegetation Red Edge	0.865	20
Band 9 - Water vapour	0.945	60
Band 10 - Cirrus	1.375	60
Band 11 - SWIR	1.610	20
Band 12 - SWIR	2.190	20

Figure 15 : Bands of sentinel 2 — extract from my handwritten notes

Other remarks on the dataset

Clever Hans effect , or why I couldn't the ortho-images across Landsvik and Vega.

Clever Hans was the nickname of a German horse in the 20th century believed to be able to solve math problems and understand language, responding for example to arithmetic questions by tapping his hoof on the ground. It was later discovered that the horse was in fact reacting to subtle cues from his human questioners — like small changes in posture when the right number of hoof taps had been reached. The *Clever Hans* effect thus refers to a situation where models derive the right answer from non desired signals.

In the task at hand, this issue arose from the fact that the classes were not spread evenly across Vega and Landsvik : take V1-C-8 for, one class of ground water mire that occurs disproportionally in Vega. During training, it would thus be shown mainly to the model in RGB. Hence, when shown one RGB image (from Vega), the model will be over-incentivised to predict the class V1-C-8, and under-incentivised to predict the class when it is a CIR image (from Landsvik). Since this class is indeed much more common in Vega than in

Landsvik, the model will have better accuracy due to the channel cues, instead of relying on features pertaining to lime-poor ground water mires.

This problem occurs very often in deep learning, for instance for image analysis of biological tissues, as they are not stained exactly the same from one hospital to another, and hospitals do not have, for instance, the same rates of cancer.

Distribution shift between Landsvik and Vega

On some bands, the distribution shift is as big between a same class from Vega vs Landsvik as between two different *hovedtyper* from Vega, as shown in the scatterplot matrix below (eg SWIR or green), which raises concerns for the ability of the model to generalise to other datasets.

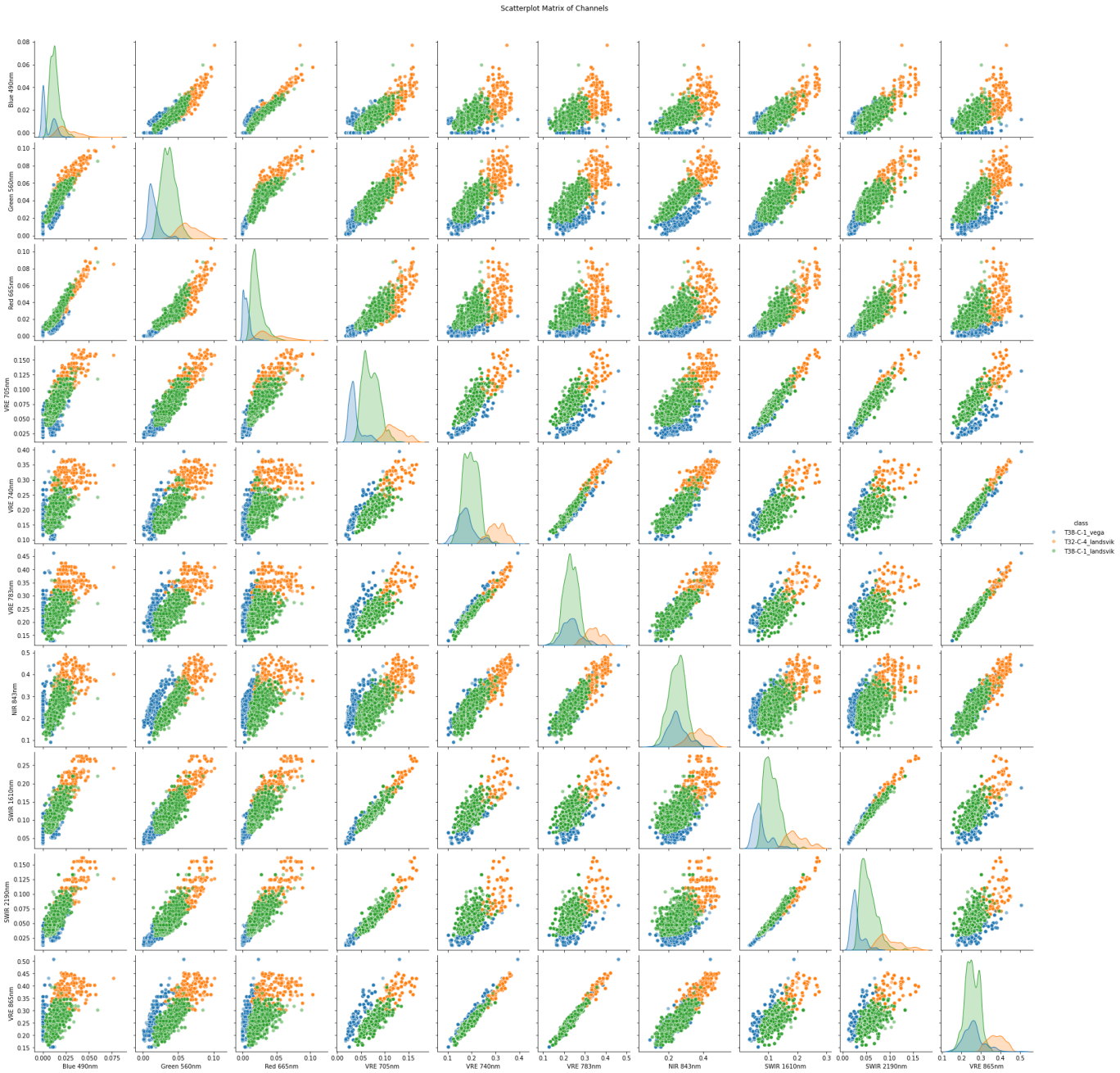


Figure 16 : Scatterplot Matrix of the distribution of one class from Vega and Landsvik and another class from Landsvik across 10 bands from sentinel 2

I. Pre-training from scratch vs fine-tuning

To train a deep learning model, one option is to initialise the parameters randomly, and then train the model until it converges to a local minimum. But this is a ressource intensive process, both in terms of data and computational power. Another option is to take a model that has been pre-trained on another task and fine-tune it on the task at hand. To break down how this can work, let's take the example of a CNN.

A CNN takes an image as an input, and runs a sliding kernel on the whole image. For each position, it calculates a convolution between the kernel (whose parameters are trainable) and a part of the image, and gradually maps the input image to a new feature map as it crisscrosses the entire picture. Then it can downsample the image (for instance through a max-pooling of every 2x2 pixel patch, or by having a bigger stride), and run another convolution on the newly calculated feature map. By so doing, it creates higher and higher level feature maps as it goes deep into the network. The first feature map will probably pick up on very low level features like edges, the next one will pick up on basic shapes, a couple feature maps into the network, pixels will activate when something akin to a cat is input, and the deeper you go the more abstract, high level features will be represented. Also, in most CNNs, the deeper you go the smaller the feature maps are, but the more numerous they are. At the end of the level, there is typically a stack of 1x1 feature map, each encoding very high level features. In other terms, the input of the image is mapped through a highly non-linear operation to a new, low dimension latent space (eg $\mathbb{R}^{100}$) where each direction is a feature of that reprojected data. One such feature could be whether the image shows a sense of *eeriness*. Another could be whether it presents a big amount of fluffy things.

The idea behind fine-tuning is that the first layers of the CNN need not to be retrained from scratch when one moves to another task, because the low level features are the same whether one deals with cats, chess boards, or wetlands. Being able to see shapes and not a mere colour jumble is useful for any task. Except at the very end (where it may be task specific), the network organises the data from a muddle of stimuli into a set of highly structured feature maps. It is akin to what we, humans, do when we learn a task : the visual processings required to be an art historian are to a large extent the same as those required to be a top level tennisman. Namely, structuring the patches of colour that fire from the retina into shapes and objects resting in a three-dimensional space.

The convenient thing is that achieving this fundamental framework to perceive and analyse the data is in fact the bulk of the computation workload when one trains a model. By using pre-trained weights, we initialise the model much closer to a good local minimum than if we set random parameters.

From here, there are two possibilities. The first, cheapest one is to merely train a small multi-layer perceptron (MLP) on the last stack of 1x1 feature map (a vector in the last latent space) : instead of training millions of parameters through 18 layers (in the case of ResNet18), it allows to train a few hundred parameters. The main drawback is that the features in that vector may be too task specific. For instance, if the model was initially trained to predict animals classes, it could be that one feature [namely, one component in the vector] takes a high value when a certain amount of fluffiness is shown. This will perhaps help our MLP tell appart a lush grassland from a sand desert (arguably, the fluffiness neurone will fire for the grassland) but it may not help to differentiate a desert from a boreal forest.

Another option is thus to fine tune the entire model, computing the gradients back through the whole network at every batch. After one try that showed me that I had enough computing resources, this is what I did in my code for both the ResNets and the SwinTransformer : below, a snippet from my code shows

param.require_grad set to true for all the model's parameter : in pytorch lingo, it means that the gradients will be calculates to update the weights for all the parameters in the model, and not just the fully connected layer at the end.

```
47  # IMPORT A MODEL
48  model = torchvision.models.resnet18(pretrained=True)
49  #model = models.swin_v2_t(weights="IMAGENET1K_V1")  #
50  model_name = type(model).__name__
51
52  # Freeze all layers or not
53  for param in model.parameters():
54      param.requires_grad = True
55
56  # tête de classification
57  model.fc = nn.Linear(model.fc.in_features, 7)
58  #model.head = nn.Linear(model.head.in_features, 12)
59
```

Snippet of code showing where to freeze or not the layers' parameters

There is however one technical hurdle in the way of applying pre-trained pre-trained CNNs to multi-spectral images.

From the input three channel image to the first layer, the feature maps are produced using a kernel of 'depth' 3. For instance, a 3x3 kernel to produce a feature map in the first layer would have 27 pre-trained parameters (cf Image below)

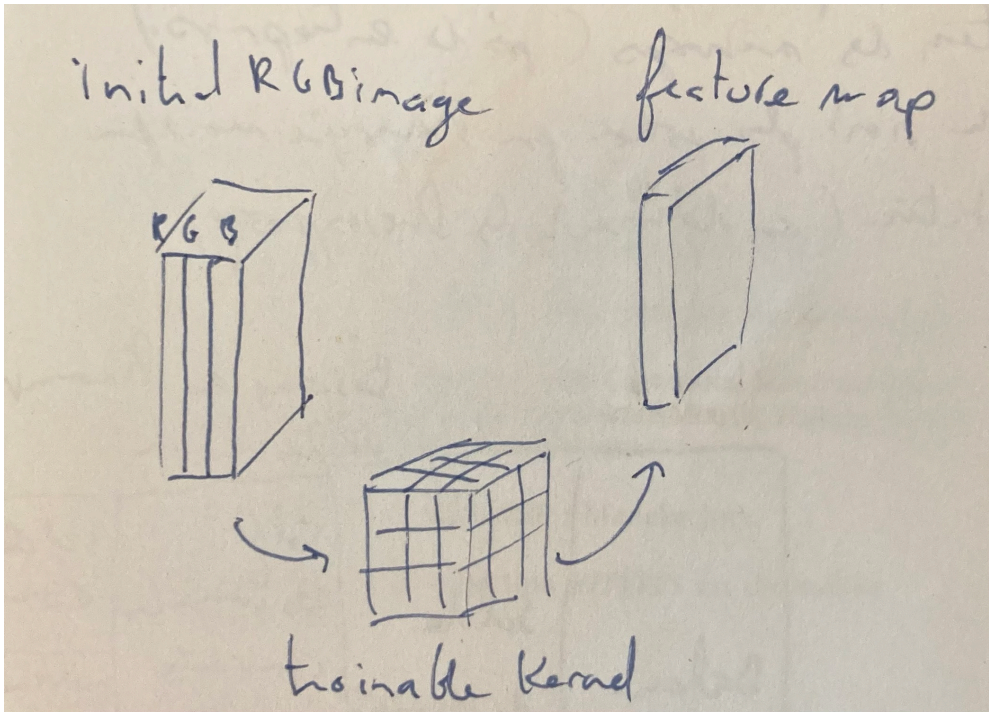


Figure 17 : CNN feature extraction diagram

But here's the catch : to handle a multispectral image, one would need a kernel with more layers. But what value should we assign to those layers ? They can't be spelled out of thin air, and we don't know how the

model actually structures the data, so if we use random initialisation it will feed gibberish to the next feature map, and all the information flow will be completely scrambled right from the start, and our model will be useless.

So the best solution would be to be able to train the model from scratch with deeper kernels between the first and second layer. But this would also require a retraining every time we change the channels we use, which is bound to be the case if we use images from different captors (and the number of possible combinations quickly explodes with multispectral data) so it would be very impractical.

I thought I could perhaps tweak the first layer of ResNet to assign the new kernels values based on those for the closest channel in the the spectrum (the kernel sliding the NIR channel would receive that of the red one), hoping that the kernels for a given feature would do not differ too much from one channel to another. Unfortunately, I printed all the 64 kernels of the 64 feature map of the first convolutional layer of ResNet (adjacent figure), and although mosts of them are very similar for all three channels, a significant number them differ widely and in unpredictable ways, as on the adjacent figure.

In the last part of this report, I will discuss ways to train the model from scratch in spite of the very small amount of labeled data.

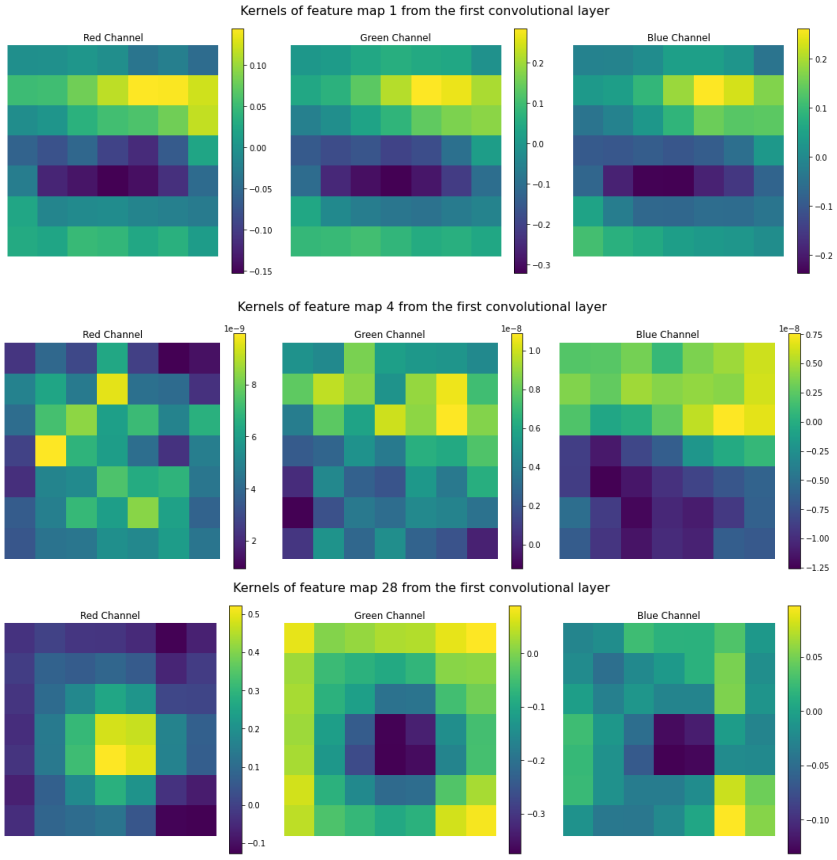


Figure 18 : Selected kernels from the first layer of a pre-trained ResNet-18

II. Transformers and CNNs

I've juste expatiated on CNNs-related considerations, but although CNNs are ubiquitous in the field of deep learning for image analysis as well as very convenient because they are easy to interpret (the deeper you go in the network, the more you abstract away information from the input), a new kind of architecture is gaining traction in the past few years : the transformers architecture. Transformers were initially devised for natural language processing, and proved ground breaking for handling long sequences by getting rid of the bottleneck in LSTMs (long short term memory) and other RNNs (recurrent neural networks). To put it simply, at some point when one use a RNN, all the meaning of the input sequence in crammed in one vector of a fixed dimension. It is not a problem when one deals with short sequences — the sentence *I eat an apple* can be properly represented by a dense vector in $\mathbb{R}^{100}$ —, but when a full text is fed to the network, there's no way to represent all the meaning in such a low dimensional space.

Transformers get rid of all recurrence and bottlenecks and convolution by using a simple '*self attention mechanism*' that I am not going to explain here, both because I don't understand the underpinnings quite well and because it's tedious, but the bottom line is that it doesn't care the length of the input sentence, and scales

only quadratically with the length the input. The *attention heads*, akin to CNN's feature maps learn representations very efficiently, and so this architecture was at the basis of the revolution in language modelling that led to chatGPT and it's likes. As it gained attention from other fields, the vision community was quick to implement it on images (to put it very simply, by flattening pictures into a very long sequence of pixels, and feeding it to the model), and it yielded state of the art results on image analysis benchmarks.
Because one of my big courses was about neural methods for natural language processing last semester, I was very eager to try one implementation of this transformer architecture. I chose the Swin Transformer, because it showed better performance and had an interesting architecture, reminiscent of the CNNs (instead of running a costly self attention on big patches of the image like ViTs do, it runs a sliding attention on a very small window to generate higher levels feature maps, on which it runs another sliding attention, and so on and so forth). In the benchmarks, Swin Transformers are supposed to yield far better results than CNNs, however, it is not what I experienced, suggesting that the limiting factor was the quality of the data, not the performance of the model — which does not come as a surprise given the very high variability in the labels from one mapper to another.

ResNet-18 vs SwinTransformer

I ran numerous experiments to tell see if SwinTransformer lead to better results than ResNet19, and I generally ended up with similar performance with both architectures, except that it took much much longer to train the Swin Transformer.

Below, I used 128x128 ortho-images labeled from Ander's polygons in Landsvik, and we can see that both models predict reasonably well most classes, and fail similarly to predict the different ground types from the class T4 — the forest main type. They both classify all subtypes as the most common one in the dataset (T4-C-5).

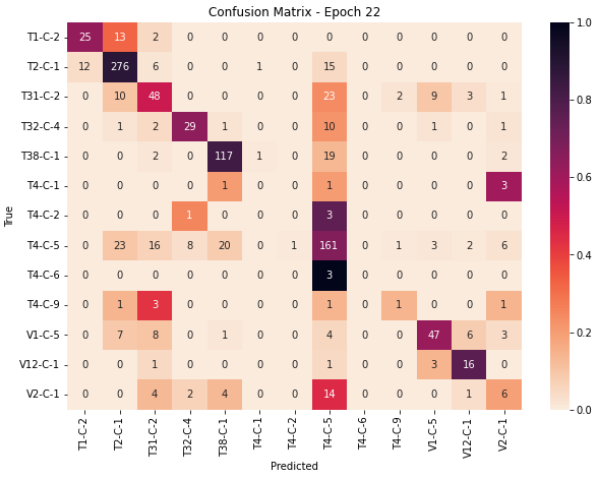


Figure 19 : Confusion matrix of the predictions on the validation dataset of a Swin Transformer fine-tuned on ortho-images from Landsvik labeled with Ander's polygons. Learning rate : 0.0001. Optimiser : AdamW. Batch size : 16

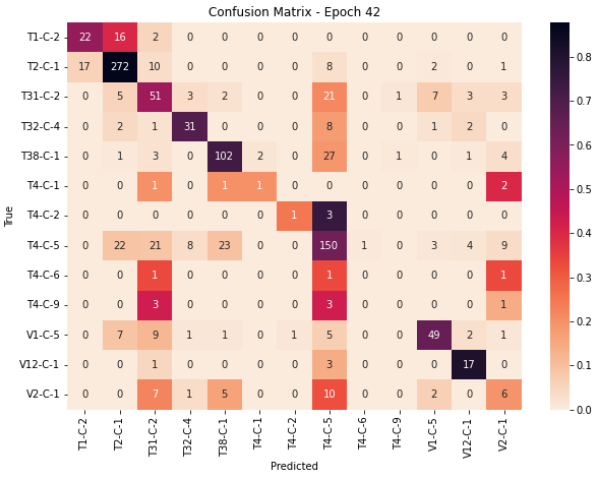


Figure 20 : Confusion matrix of the predictions on the validation dataset of a ResNet-18 fine-tuned on ortho-images from Landsvik labeled with Ander's polygons. Learning rate : 0.01. Optimiser : AdamW. Batch size : 16

After these experiments, I only used ResNet-18 because it was much much quicker to train, and it was easier to make it converge (the Swin Transformer required to adjust gradually the learning rate to get good predictions, and I couldn't use my computer when it was being trained).

III. Results from across different datasets : ground pictures, drone images and ortho-images

a) Some general technical details

PyTorch

I used PyTorch as my deep learning framework. I downloaded the pre-trained weights from torchvision.models.

Splitting the development dataset.

I always split my development dataset into a training dataset on which the model is trained, a validation dataset on which it is tested at every epoch, and a test dataset. I usually give 80% of the data of each class to the training dataset, and 10% to the validation and the train. In my code, I check that the intersection is zero to avoid any leak.

In practice, I never used the test dataset because instead I averaged the metrics across different epochs (I believe the test dataset is useful to prevent overfitting on the validation dataset, eg is one selects the best model out of 50 epochs, it may be that this specific one overfits the validation dataset, but this concern is immaterial if one only cares about the average results).

Data augmentation

Because the data is very scarce, I systematically used some of the data augmentation techniques from torchvision.transforms. I used very small ColorJitter, in some cases RandomCrop, RandomHorizontalFlip, RandomVerticalFlip, and RandomRotation.

What optimiser to go down the loss landscape ?

I used AdamW as my optimiser. To put it simply, AdamW (like Adam) keeps track of the past gradients for each parameter in order to have bigger learning rates in the directions (=parameters) where the gradients are smaller (so it takes a big step if it's not very steep, but treads carefully if it *is* very steep to prevent overshooting). It also has some inertia, meaning that if in one direction, there is a plateau, it will keep moving for a while in the direction that used to be the direction of steepest descent (this also allows it not to get stuck in small ditches in the loss landscape)

Metrics

It does not appear on all figures because I enriched my code throughout the weeks, but in most cases I calculated the ***average macro averaged F1 score over the last 10 epochs.*** Lets break this down : for a given class, the F1 score is the harmonic mean of precision and recall. Precision means TP / (TP + FP) : if you fish for salmons, it is the number of salmon out of everything you got from the lake. Recall is TP / (TP + FN). It is the same as *true positive rate*, and the same as *sensitivity*. It is the amount of salmons that you fished out of all the salmons in the lake.

Now what is the macro-averaged F1 score ? When you have multiple classes with different amount of instances, and you want to sum everything in a single metric, you can either average the F1 scores across all classes, irregardless of the number of instances in them — this is macro averaging —, or calculate a weighted average based on the number of instances : this is micro-averaging. Because I had very imbalanced class and didn't want two of them to take over the whole results, I calculated macro-average F1 score, although this was probably a debatable choice in some instances.

Because the macro-average F1 score fluctuates from one epoch to another, at every epoch, I calculated a sliding average of the past 10 epochs. Hence the ***average macro averaged F1 score over the last 10 epochs***

b) Leveraging the ground images (with the consensus points from Landsvik)

The good thing with that dataset is that the labels were from the consensus points, so they were more consistent than the other labels I used in the other experiments. However, it is the data, not he labels that were inconsistent : some pictures were taken vertically close up to the ground, whereas other ones were taken horizontally with some vegetation meters away and featured patches of sky. It was even more difficult to use considering the very high resolution (about 3000x4000), which was way way too big to input models like ResNet or ViT that typically take 224x224 images — in the case of resnet, the field of view would clearly not reach the extent of the image, making it impossible for the model to capture long distance dependencies.

To deal both with that issue *and* with the scarcity (for some classes, I had only one image left for the validation dataset), I used RandomCrop of 224x224. However, this did not solve everything : even if I make crops from one image, the model is shown data with very small variation since all the crops come from the same image, which make it difficult for it to generalise to the validation dataset that necessarily presents distribution shifts from the train dataset. Also, because some images were taken facing the horizon instead of the ground, in some instances the model probably ended up being shown a patch with nothing else than blue sky, which it added a lot of noise in the data.

Therefore, it doesn't come as a surprise that the results were bad, especially for the tree classes in the middle (cf adjacent confusion matrix below) for which there were only one or two instances in the validation dataset from which the matrix is derived. The macro averaged F-score is heavily burdened by these three classes where no right prediction is made : here it would have been smarter to calculate a Micro averaged F score !

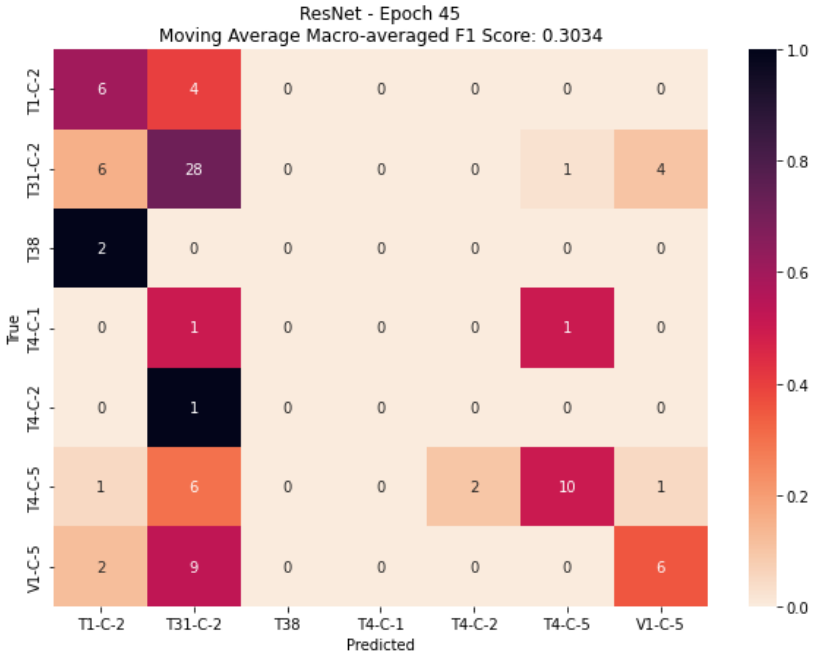


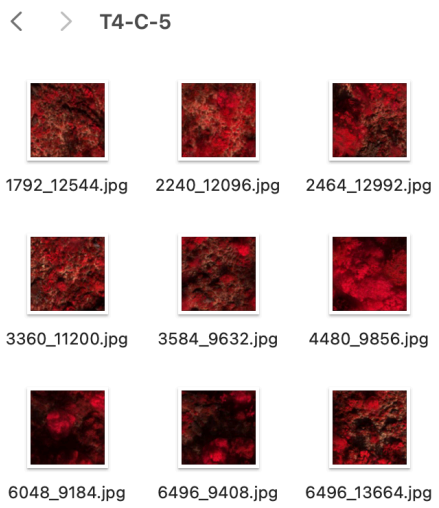
Figure 21 : Confusion matrix of the predictions on the validation dataset of a ResNet-18 fine-tuned on ground level pictures labeled with the consensus points dataset.

Deep down, I cant rule out that this approach may work if there were sufficient data, but there was way too few pictures and they were very inconsistent. It's hard to assert any conclusion as to whether this is a good lead of inquiry, because the fact to the matter is that this approach could be tested only on the four classes for which there was more than 30 images, and they all belonged to a different *hovedtyper* (level 2), so we cannot even assert that the ground images help telling appart classes at the *grunntyper* level (level 3).

c) Leveraging the drone images (with Ander's polygons from Landsvik)

Because there was way to few data across just 4 usable classes in the first dataset, it decided to use the polygons from single mappers instead of the consensus points. I wrote a python code to cut the big tiff file of landsvik in 224x224 patches, and then wrote another code to assign a label to each patch if it fell within one (and only one) polygon. This strict condition probably produced a slightly more consistent dataset, because the patches were less likely to be very close to the border of a polygon (since if they crossed it in one single pixel they were not kept). And since the polygons boundaries are contentious between the mappers, it is nice not to show the model patches that are too close to the edges of the polygons.

This approach offered much more instances per class (at the cost of a probably less consistent dataset), because it was not just points that were labeled, but entire areas. So for each class I ended up having many, many patches. Many new classes showed up in this dataset, although once again, some had very few instances. But as a general rule, all the classes that *existed* in the consensus dataset, had enough instances to train in this new one.



Screenshot showing some of the patches of forest obtained by intersecting the labeled polygons with the drone images. It shows on the of forest subclass.

To have some basis for comparison, I trained the model on the same 7 classes as in the previous experiments, and the results were way, way better, as shown in the confusion matrix.

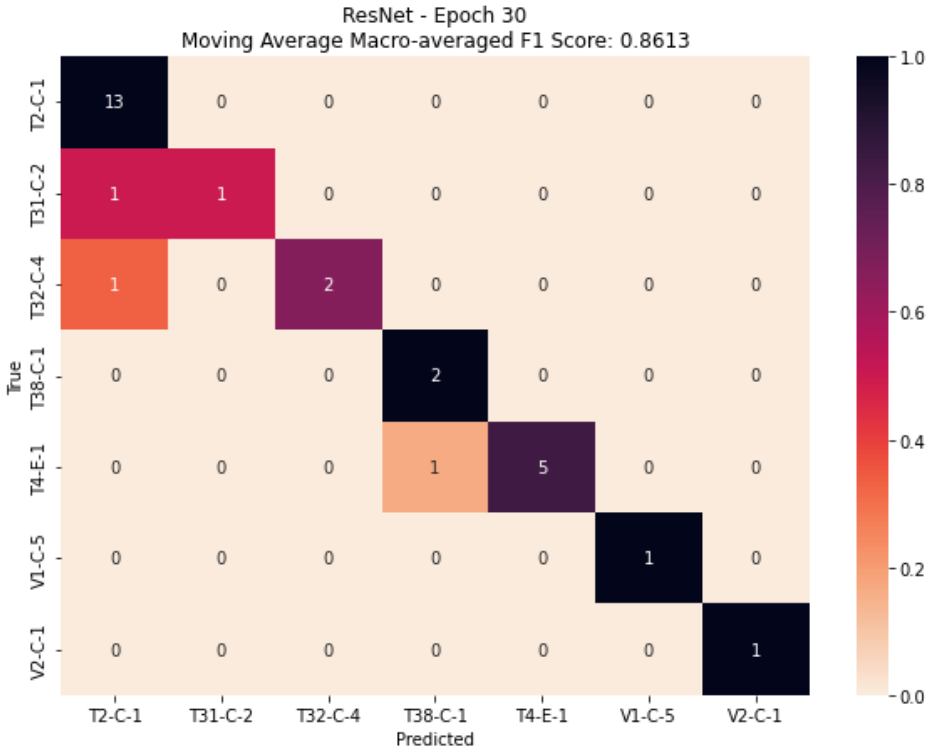


Figure 22 : Confusion matrix of the predictions on the validation dataset of a ResNet-18 fine-tuned on 224x224 drone images labeled with Ander's polygons from Landsvik

The model quasi-certainly overfits the development dataset, and would perform much worse if tested on data from another location

The average F1 score topped 0.86 on average between epoch 20 and 30. This is a bit suspicious because it is beyond the variability between two mappers (how come does the model predict the classes so well, when two mappers would have labeled them differently 60% of the time ?)

It probably stems from the fact that the patches were labeled from just a handful of places, dozens of meters appart, so the model learns something that has more to do with that specific place, than some generalisable representations of the classes. To get the bottom of it, one would need to test the model on data from another location, but I couldn't do it for two reasons :

(i) Label side : most of the classes in Landsvik are not present in Vega

(ii) Data side : even on the intersecting classes between Landsvik and Vega, I couldn't do it because I do not have drone data for Vega, and the ortho images do not use the same bands

However, I did this on the 5 most frequent common classes between Vega and Landsvik using sentinel 2 data, as presented in part III.

d) Moving on to the ortho-images, and shrinking the image sizes.

At first, I didn't notice that the ortho-images did not use the same channels in Vega and in Landsvik, so I thought that I could use them to use the full dataset, across both location, to test the approach against much more classes. The problem is that the ortho-images have a much worse ground resolution than the drone images, and so if I took 224x224 patches, they would have a much bigger spatial extent than with the drone (from 7 meters per 7 meters to 28 meters per 28 meters). With such a big extent, many patches would be to big to fit any polygon, and so I wouldn't get instances in many classes. So went on to see how the model performed when I shrinked the size of the images.

The adjacent figure is the result using 32x32 patches, and the pixel have a 12.5 cm ground resolution (vs 224x224 with 5.5cm ground resolution in the result above). Unsurprisingly, it doesn't perform as good : 0.51 macro averaged F1 score, vs 0.86, but the result is affected by the V2-C-1 class that the model didn't pick up on at all. So it is still an interesting result considering how much coarser the training data was (1024 pixels per image vs 50175).

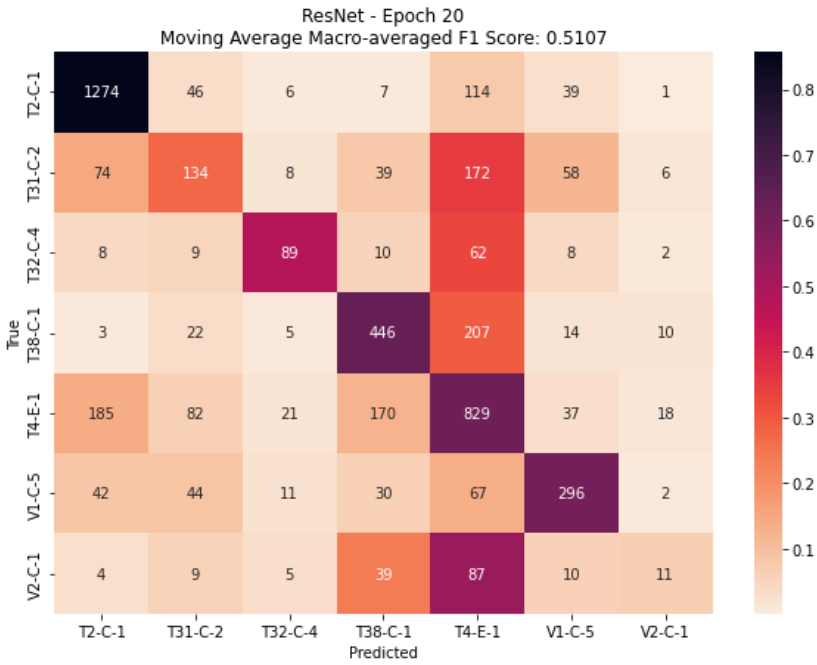


Figure 23 : Confusion matrix of the predictions on the validation dataset of a ResNet-18 fine-tuned on 224x224 ortho-images labeled with Ander's polygons from Landsvik

A MORE SUCCESSFUL ENDEAVOUR : A MERE MULTILAYER PERCEPTRON ON HYPER-SPECTRAL SATELLITE IMAGES

I. Class prediction with a MLP and sentinel 2 data

a) The pros of sentinel 2 data

Since I noticed with the CNNs that shrinking the size of the images didn't worsen that dramatically the F1 score, I assumed that the main cues came from the spectral signature. This was a convenient conclusion, because it meant it was a good idea to try and use multi-spectral sentinel 2 images instead. Because the ground resolution of a sentinel 2 pixel is about 7 per 7 meter, I could get rid of the *'image'* thing altogether, ie the shape and texture information, and only feed a model with the average spectral signature of a 7x7m patch of land. With this approach, I expected more interesting results for many reasons :

(i) The spectral data had seemed to be much more informative than shape and texture on ecosystem type prediction

(ii) 7x7 meter is a handy size with respect to the polygons size, as it generally shows a representative snippet of the nature type.

(iii) The data is available for everywhere, and there are some cloudless days every spring, so it could later be used to keep track of land use changes. More specifically, I finally had the possibility do train the model on data from both Landsvik *and* Vega, and those see how it scales with more data (this should improve the F1 score) and more classes (this should worsen the F1 score).

(iv)

This was also a handy perspective because it meant I wouldn't need a CNN, as a mere multi-layer perceptron would be fit for the task.

b) What is a MLP ?

Multi layer perceptron is the (fancy) name of a fully connected network that traces back to the early ages of AI, when researchers in cybernetic believed they were about to crack the brain's mechanism. But it simply means a small network, with say, 2 hidden layers that reproject the input data to a new latent space. On the adjacent drawing, it is fed with the three values of a RGB pixels, that are reprojected to a 5 dimension space (first hidden layer), then to a 4 dimension space (second hidden layer), and outputs logits (activation rates) for two classes.

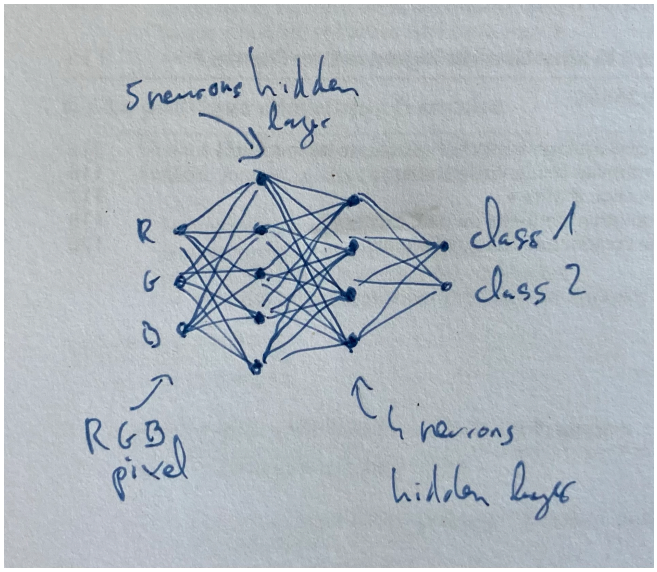
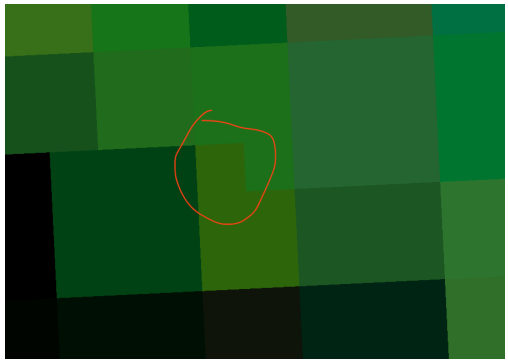


Figure 24 : Drawing of a simple MLP

I did the same, feeding the network with 10 of the 12 sentinel 2 bands, and trying different sizes (varying the number of hidden layers, and the number of neurons per layer).

c) Building the dataset, and avoiding data leak between the train and validation set

I used the same approach as before, simply intersecting sentinel 2 pixels with the shapefile from Anders mapping. At first I was amazed by the results, but then I realised that there was a huge leak of data between, the train dataset and the validation dataset. The reason comes from the fact that the resolution is not the same for all the bands of sentinel 2: while blue, green, red and nir had pixels about 7 meters wide, the 2 SWIR bands and the 4 red-edge bands had double the width. It meant that many pixels were very, very similar, except for the 4 seven meters bands. In the adjacent image, you can see that actual resolution is bigger than meet the eye at first. So if it randomly distributed pixels in the train and validation dataset, it was certain that many pixels would be found in both dataset.



Clue of the oversampled images (screenshot from Qgis)

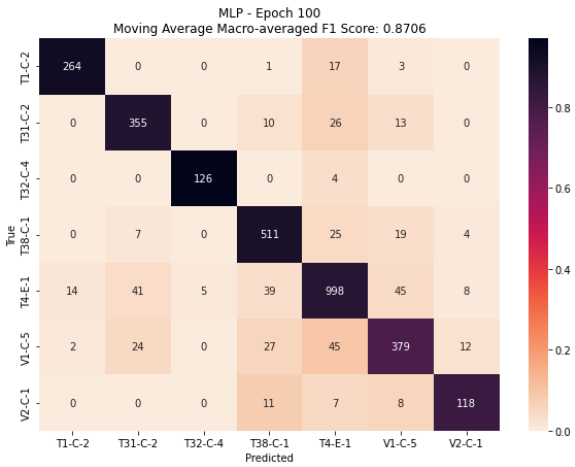


Figure 25 : Stellar results before I realised there was a massive leak.

I therefore wrote a code to get rid of all the duplicates in the development dataset, and re-run some of my experiments. My data set shrank by… 75 %. The performance got worse, but was still fairly decent.

d) Results & comments

Technical choices

I standardised the inputs at the pixel level (meaning that the average values across all the bands for **every** pixel was 0, and that the standard deviation was 1). I was surprised but it yielded significantly better results. It is perhaps because it prevents one single feature with big values all the time (some band on which it's generally brighter) to dominate the activations in the network. It may also help with gradient descent if the gradients don't vary to much (as they would do if features are on very different scales).

Do prevent overfiting, I used a big dropout (0.2)

I tried with different sizes. The bigger it was, the better it went, but it stared to plateau around 512-256 (I used two hidden layers). I opted for smaller architectures, because even if I didnt overfit the training dataset, I probably overfit the validation dataset, and I thought that the model was more likely to learn generalisable representation if it wouldn't come up with too complex ones. So it mostly sticked to 128-64 (first layer with 128 neurons, second with 64). Assuming 12 output classes, as in the confusion matrix below, it still meant 10144 parameters !

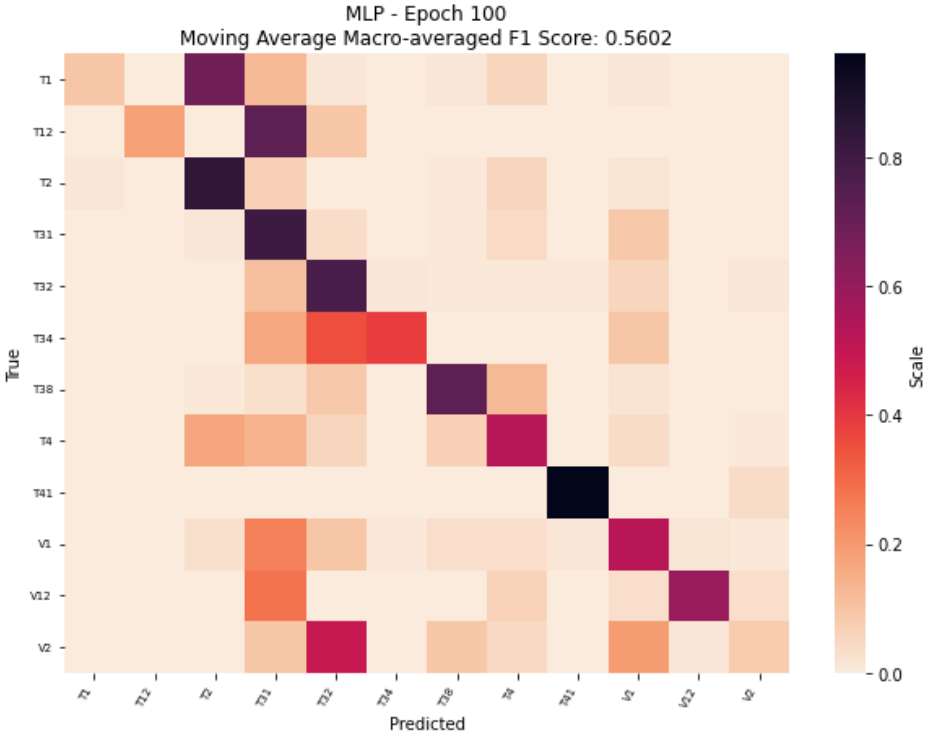


Figure 26 : Confusion matrix of the predictions on the validation dataset of a 128-64 MLP for 12 classes at the *hovedtype* level across both Vega and Landsvik.

Figure 27 : Evolution of training loss, validation loss, and Marco F1 score in through the training of the MLP

At first I didn't understand how the training loss could be higher than the validation loss (chart above), and it had never been the case with the CNNs, but I figured out it was because I used a big dropout during training, and not dropout not during validation, so the performance during training is underestimated.

I tried to see how the model managed with all the classes that I could feed it with at the grunntyper level (level 3). Unsurprisingly, it struggled to tell apart *grunntyper* from the same *hovedtyper*, but the overall result was surprisingly good (the macro-averaged F1 score is bad because of the classes were F score is zero, but on many classes it is good), although I don't think it's generalisable (I think it merely overfits the development dataset).

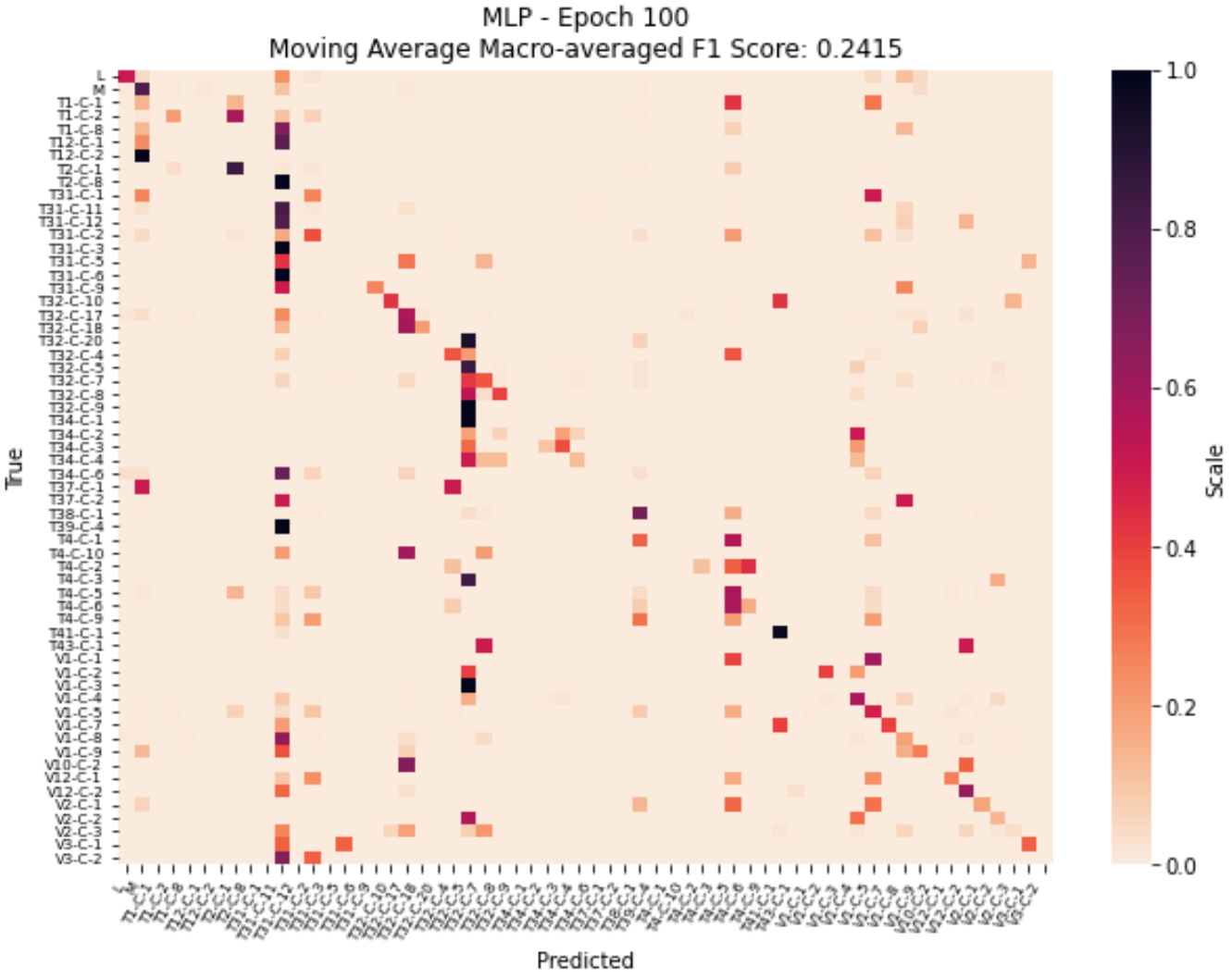


Figure 28 : Confusion matrix of the predictions on the validation dataset of a 256-128 MLP for all classes at the *hovedtype* level across both Vega and Landsvik.

I believe the models overfit the development dataset because there is still some sort of leak, in the sense that even if the model cannot learn exactly the dataset (because there is no pixel that lies both in the training and the development dataset), since the changes in an image are generally continuous (two pixels next to each other have similar spectral distribution), and since I randomly assigned different pixels to the validation and training datasets, the model was validated on data very similar to what it had seen during training, making it unlikely to be robust to distribution shift if shown data from another location.

To face the truth about the generalisation capabilities of the model, I selected the few classes that occurred both in Vega *and* Landsvik (5 classes for which I deemed I had enough data). I trained the model on Landsvik data only, and tested it on Vega data only. I didn't expect a good result, because I didn't have much instances for these 5 overlapping classes, and it doesn't come as a surprise that a model has bad generalisation capabilities if it is trained on only one location : the distribution shift when it is shown data from another is

bound to be important. As shown in the confusion matrix, it yielded mixed results. For three of the five classes, the F1 score is good, but for T1-C-2 (one kind of bare ground) as well as V1-C-1 (one kind of open groundwater mire), the model was simply incapable of linking the Landsvik data to the Vega data. The model shows some generalisation capabilities, but it is hard to tell how it would scale with more classes, nor how it would perform on ground types (level 3) from the same main type (level 2).

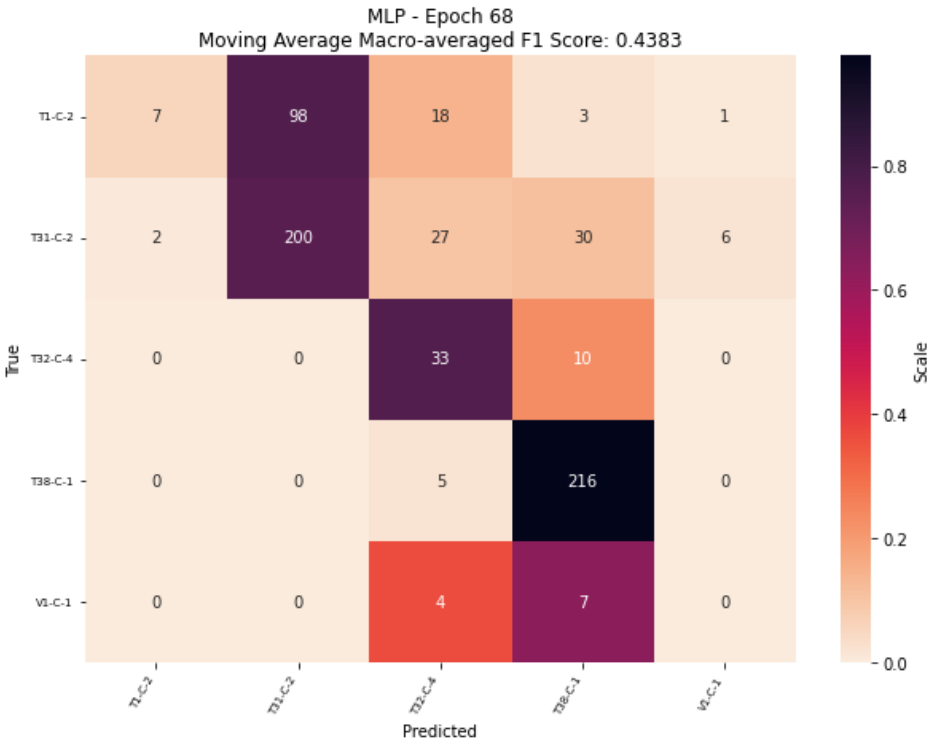


Figure 29 : Confusion matrix of the predictions on the validation (Vega) dataset of a 128-64 MLP trained on Landsvik data for 5 classes

II. A failed attempt to leverage Sentinel 1

I tried to use sentinel 1 data to tell appart ecosystems at the highest classification level (the first split separates wetlands (V types) from drylands (T types), that is areas where the soil is water-saturated all year round, vs the areas where it isn't. However, distribution of V and T types was exactly the same on the C band of sentinel 1, as shown below. Unsurprisingly my model wasn't able to tell them appart.

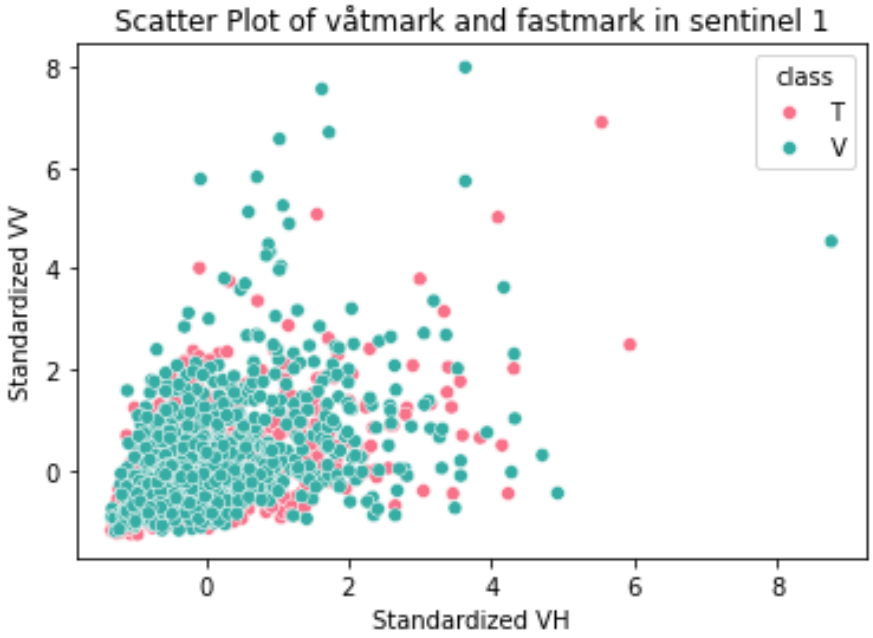


Figure 30 : Distribution of both wetland and dryland points across the VV and VH radar bands of sentinel 1

I realised the problem was more complicated than I thought : I used Sentinel 1 images from one specific day in the spring, and it could well be that it had rained on a dryland just before, temporarily saturating the generally non saturated soil. Also, the C band does not penetrate that much in the soil (just a few centimetres), so I would rather have had access to the L band, which penetrates by a few dozens of centimetres, but I couldn't find a satellite carrying such a sensor with open data.

III. Predicting lime richness

Devising a dataset

The last thing that I did was trying to predict lime richness from the sentinel 2 data. It is a very common LKM (local complex gradient) for the environmental spaces in which the ground types (level 3) are delineated. So instead of directly predicting the ground type, it is interesting to try and leverage the hyperspectral data to directly aim at these complex gradients that are ubiquitous in the NiN framework. It would enable to abstract away higher level features, for instance by lumping together forests and grassland that share the same lime-content (through some fancy computation akin to a very complex NDVI across 10 bands). It's a way of forcing the model into *thinking* like the NiN system is devised : instead of figuring out the ground type with arcane data processing, we can directly ask it to predict the lime richness. It's a bit of a litmus test for the sentinel 2 approach, because if it figures out the ground types, it has all the information it needs to figure out the lime richness (ground type ⇒ lime richness), so the contrapositive is that (unable to predict lime richness ⇒ unable to predict ground type) : if it fails in predicting the lime richness, it logically can't predict the ground type.

To that purpose, I looked up the lime content associated with each of the ground types for which the LKM is used, for both Vega and Landsvik. I ended up with letters ranging from A to I. I assigned a number to each letter (from 1 to 9). When one *hovedtype* (level 2) used coarse descriptions (like ABCD, and EFGH), I averaged them. I ended up with the dataset distribution represented in the stacked barplot below.

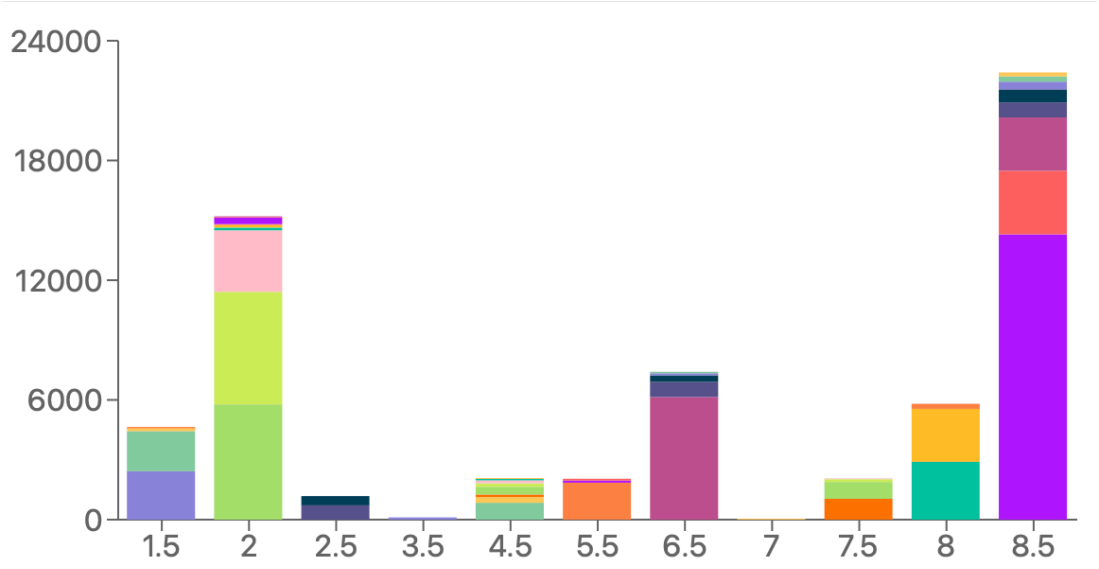
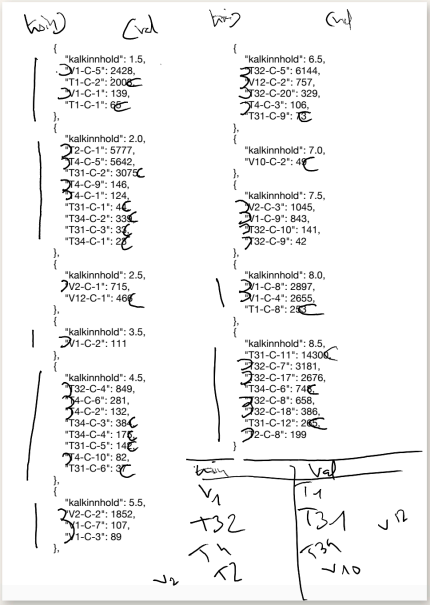


Figure 31 : Stacked barplot of the distribution of lime richness in the dataset. The colours show how much one value is dominated by a class.

Bias-proofing it



Snippet from my drafts to devise the dataset

I was afraid that, under the hood, the model reconstructs the classes to predict the lime richness : if I showed it only forests as a lime poor example, it would probably simply learn that the spectral signature associated with forest spells poor lime content, and then wrongly classify the lime rich forest in the validation dataset. So I took care to devise a dataset that contained most of the samples in the training section, but where all the subclasses of the maintypes (all the *grunntyper within a hovedtyper*) appeared in the same subfolder of the dataset (train xor validation). That way, the model would be forced, for example, to use representations learned from forests to predict the lime content of a wetland area.

Results

I trained a MLP on a regression task. It had three hidden layers (64-128-64). The results shown below show that it didn't really pick up on lime-content. The mean square error converged around 6.5, which is better than random (I ran a random predictor on the validation dataset, and it got 14) . It means that when it makes a prediction, it is on average 2.5 units away from the right value (vs 3.7 with a random predictor), which is clearly not a sound basis for predicting *grunntyper*. But it's still better than average, so perhaps it would work better with more data — and higher spectral resolution.

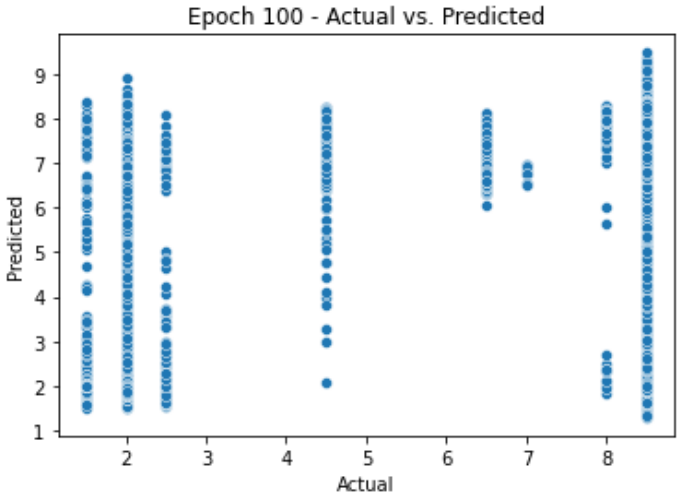


Figure 32 : Actual vs predicted lime content by a MLP trained on a regression task

However the dataset it not big, so the better than random performance perhaps simply comes contingencies (eg if most forests in train the dataset have high lime content, and in the val dataset there's another ecosystem type with a similar spectral signature, which, by chance, also has a high lime content).

CONCLUDING REMARKS

The results leave me with a twinge of disappointment because they fall short of my expectations. Nothing guarantees that the models are learning generalisable representations, and where the results are good, they are suspiciously so, given how inconsistent the labels are. This is especially concerning since I had to rely mainly on polygons from single mappers rather than consensus points, as I simply didn't have enough data otherwise. However, after grappling with this problem for 11 weeks I can cautiously draw draw several conclusions and suggest some insights about what paths to explore next

The quality of the labels is probably the limiting factor

I believe that the quality of the labels is the limiting factor (otherwise, bigger models, or better models — like transformers) should perform better than smaller ones. It probably doesn't come as a surprise since we already knew how much inter-mapper variability there is. But it also points towards choosing architectures that are both simpler and less computationally heavy.

Hyperspectral data could help a lot

Since spectral data appears to offer more cues than spatial data (shapes and textures) at least for the predictions at the *hovedtype* level, it would be worth looking into other hyper spectral satellites could be used It is not very clear to me whether this data is accessible for free, but Germany recently launched an hyper-spectral satellite called enMAP with a whopping 242 channels — a far cry from the 12 bands of Sentinel 2: if someone finds a way to get hold of these data, it could be incredibly useful !

Radar data

Sentinel 1 only has C band, but L band penetrates much deeper into the soil. L band, paired with bands taken at a moment where we know the soil is unusually dry would perhaps be efficient to spot the wetlands.

Lidar data

I neither had the time, nor the computing power to process them (I couldn't even open the file), but the lidar data acquired is probably very very rich, and is worth being leveraged, either rasterised, or directly (many open source models using point cloud directly as an input are being investigated in the past few years).

Texture and shape for the ground types

If there were more data, and if the ground photos were taken in a more consistent way, it would probably be useful to figure out the LKMs, as for instance one could expect a good transformer to pick up on calcicole plants if the resolution is good enough.

Contrastive learning to solve the CNN on multispectral data problem

Drone images with both high ground resolution and more than 3 channels will also very precious data when there will be more labels, it will however require to adapt a CNN to accommodate more then 3 bands, perhaps

by pertaining it from scratch through contrastive learning. It is a non supervised learning technique that consist of showing pairs of data to a model. Some of the pairs are altered version of a same image, and some are different images. The model is trained to telling whether it is a 'true' pair or not. By so doing, it learns the low level features (and some high level representations) without the need for any labels. It requires much more computational power since it means training from scratch, but the HPC service at University of Oslo (eg Fox) is totally fit for the task.

Although I do not think that my results are very conclusive, nor of any immediate use, it has honestly been an amazing internship. I was delighted with the workplace environment, and truly enjoyed the company of my GEco fellow workers. Even if I had a hard time to deal with the scarcities and inconsistencies of the dataset, it was already awesome simply to be able to put immediately into practice many of the things that I had been taught in the previous semester. The lectures were very much centred on the linear-algebra underpinnings, and I feel like I now have a much more practical hands on knowledge of how to train vision neural networks, as well as the many biases these techniques are liable to. It was also in the continuation of the amazing *Ecological Climatology* class my tutor was teaching in, and I loved to learn about the ecological processes that control the landscapes. Even more so, I loved to be taught the norwegian names of tiny calcareous flowers, and *Gyldendals nordiske feltflora* will from now on always follow me in Scandinavia.

*

Addendum on Corporate Social Responsibility at GEco (requested paragraph)

On the social dimension of CSR, I could witness that GEco fully aligns with the strong Norwegian trade union culture, and promotes a very safe, non-judgmental workplace environment.

When it comes to the environmental dimension, the research carried out in this group is evidently linked to the ongoing climate and ecological crises, as it consists of mapping ecosystem types, which in turn gives a fine-grained representation of how the carbon cycle varies in space. Every year, we emit circa 40 GT CO2, and about half of it is sucked up by carbon sinks, 9.5 GT of which ends up in terrestrial ones, namely the soil and the biomass. However, over the past decade carbon sinks have been losing pace with our emissions (reacting to higher CO2 concentration, they grew bigger and bigger in the past century, but not as quickly as our emissions), and ominously, they brutally collapsed to a an apocalyptic 2 GT in 2023 according to a preprint (*Piyu Ke & all, 2023*) widely shared in the French press[1] although the olympics somehow still managed to hog the limelight. No one had anticipated such a dramatic collapse, and since the carbon cycle is barely taken into account in the climate models, this piece of information looms very large for the trajectory of the earth system in the 21st century. To put it in very very crude terms rather than forbiddingly academic language, the climate system is going to hell in a handcart, and body bags, not polar bears, should embody the ongoing crises.

Hence, the generous interpretation is that any research about ecosystems and the carbon cycle is important, and you can't have too much of a good thing, can you ? But there's a more cynical one that I hold just as true : what's the point of mapping nature types when policy makers override all common sense and support the building of highways cutting through wetlands, like the infamous A69 in south-western France ?

One could argue that if the conditions for human life on earth are to be preserved, activism, not research, should be the way. Joining forces, hands on deck and bodies on the line, to burn down excavators along the A69 route might be our best chance to be able to grow food in 50 years !2

[1] https://www.lemonde.fr/en/environment/article/2024/07/30/terrestrial-carbon-sinks-collapsed-in-2023_6704503_114.html

[2] In February 2019, 1000 scientists called in Le Monde to join civil disobedience to face climate emergency, forced to acknowledge that legal avenues had failed us.

BIBLIOGRAPHY

Marschner, F. J. (1950). Major land uses in the United States (map scale 1: 5,000,000). *USDA Agricultural Research Service, Washington, DC*, *252*.

Vali, Ava, Sara Comai, and Matteo Matteucci. "Deep learning for land use and land cover classification based on hyperspectral and multispectral earth observation data: A review." *Remote Sensing* 12, no. 15 (2020): 2495

Thapa, Aakash, Teerayut Horanont, Bipul Neupane, and Jagannath Aryal. "Deep learning for remote sensing image scene classification: A review and meta-analysis." *Remote Sensing* 15, no. 19 (2023): 4804.

Ullerud, Heidrun A., Anders Bryn, Rune Halvorsen, and Lars Østbye Hemsing. "Consistency in land-cover mapping: Influence of field workers, spatial scale and classification system." *Applied Vegetation Science* 21, no. 2 (2018): 278-288.

Zhao, Shengyu, Kaiwen Tu, Shutong Ye, Hao Tang, Yaocong Hu, and Chao Xie. "Land use and land cover classification meets deep learning: a review." *Sensors* 23, no. 21 (2023): 8966.

Demir, D. B., and N. Musaoglu. "Automatic Classification of Selected Corine Classes Using Deep Learning Based Semantic Segmentation." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 48 (2023): 71-75.

Pollatos, Vasilis, Loukas Kouvaras, and Eleni Charou. "Land cover semantic segmentation using ResUnet." *arXiv preprint arXiv:2010.06285* (2020).

Bryn, Anders, Guri Sogn Andersen, Trine Bekkby, Harald Bratli, Børre Kind Dervo, Margaret Dolan, Rune Halvorsen et al. "Hovedveileder for feltbasert kartlegging. Terrestrisk, limnisk og marin naturvariasjon etter NiN (3.0)." (2023).

Bratli Harald, Anders Bryn, Anette Edvardsen, Lars Erikstad, Rune Halvorsen, Peter Horvath, Trond Simensen, Børre K. Dervo, and Olav Skarpaas. *Natur i Norge: Variasjon Satt i System*. Oslo: Universitetsforlaget, 2024.

Ke, Piyu, Philippe Ciais, Stephen Sitch, Wei Li, Ana Bastos, Zhu Liu, Yidi Xu et al. "Low latency carbon budget analysis reveals a large decline of the land carbon sink in 2023." *arXiv preprint arXiv:2407.12447* (2024).

Bonan, Gordon. *Ecological Climatology: Concepts and Applications*. 3rd ed. Cambridge: Cambridge University Press, 2015.

Glossary and Useful Acronyms

LKM (Local Complex Gradient):

A gradient representing environmental factors (like lime content, soil moisture) that influence species distribution within an ecosystem. LKMs are used to delineate ground types within the NiN classification system.

MLP (Multi-Layer Perceptron):

A type of artificial neural network that consists of multiple layers of neurones, where each layer is fully connected to the next one. MLPs are used for tasks like classification and regression in machine learning.

NIR (Near-Infrared):

A region of the electromagnetic spectrum with wavelengths just beyond visible light. NIR is often used in remote sensing to assess vegetation health and soil properties.

Environmental Space:

A multidimensional space in which different environmental variables (like temperature, moisture, lime content) are represented. Species distributions and ecosystem types can be analysed within this space.

Grunntyper (Ground Types):

The lowest level in the NiN classification system, representing specific ecological types based on detailed environmental conditions like soil composition or moisture levels.

Hovedtyper (Main Types):

The middle level in the NiN hierarchy, these are broader ecological categories defined by dominant environmental processes that shape species distribution, such as flooding or soil acidity.

Hovedtypegrupper (Main Type Groups):

The highest level in the NiN classification, grouping ecosystems into broad categories like wetlands or drylands based on overarching environmental characteristics.

ResNet (Residual Network):

A type of deep neural network that uses residual connections to allow for the training of very deep networks. ResNets are widely used in image recognition tasks due to their ability to overcome the vanishing gradient problem.

Spectral Signature:

The specific pattern of reflectance or absorption of light across different wavelengths for a given material. In remote sensing, spectral signatures are used to identify and differentiate between various types of vegetation, soil, and other surface materials.

SWIR (Short-Wave Infrared):

A portion of the electromagnetic spectrum that is beyond visible light. SWIR is often used in remote sensing to assess moisture content in soils and vegetation, as well as in geological mapping.

Swin Transformer:

A type of transformer model adapted for vision tasks. It uses a hierarchical structure that operates on non-overlapping image patches, allowing it to capture both local and global features efficiently. Swin Transformers have shown state-of-the-art performance in image classification and object detection tasks.

VH and VV (Vertical-Horizontal and Vertical-Vertical Polarization):

Types of radar signals used in remote sensing to measure the Earth's surface characteristics. VH and VV refer to the orientation of the radar waves when transmitted and received, which can provide different types of information about surface structures.

Red Edge (Sentinel-2)

A set of band in Sentinel-2 satellite imagery that captures wavelengths between the visible red and near-infrared parts of the spectrum. It's used to assess vegetation health, as plants reflect red-edge wavelengths when they are healthy.

CNN (Convolutional Neural Network)

A type of deep learning model designed for processing structured grid data like images. It uses convolutional layers to automatically learn and extract spatial features, making it highly effective for image classification and recognition tasks.

NEP (Net Ecosystem Production)

The net amount of carbon accumulated in an ecosystem, calculated as the difference between Gross Primary Production (GPP) and the sum of autotrophic respiration (Ra) and heterotrophic respiration (Rh).

NPP (Net Primary Production)

The amount of carbon absorbed by plants during photosynthesis minus the carbon they respire for energy (Ra). NPP represents the net carbon gain that contributes to plant growth and biomass accumulation in an ecosystem.

GPP (Gross Primary Production)

The total amount of carbon dioxide that plants capture from the atmosphere through photosynthesis. It represents the initial step in the carbon cycle within ecosystems.

Ra (Autotrophic Respiration)

The process by which plants and other autotrophs respire, using part of the carbon they capture to produce energy. It reduces the amount of carbon stored as biomass.

Rh (Heterotrophic Respiration)

The process by which decomposers (like bacteria and fungi) break down organic matter, releasing carbon dioxide back into the atmosphere. It's a key part of the carbon cycle.

Clever Hans Effect

A phenomenon where a model or subject appears to learn a task but is actually responding to unintended cues or patterns in the data, leading to misleading results.

Loss Landscape

A conceptual surface representing the loss (or error) values across all possible parameter configurations of a model. The goal of training is to navigate this landscape to find the lowest points, which correspond to the best model parameters.

Batch

A subset of the training data used in one iteration of model training. Processing data in batches rather than all at once helps to stabilize and speed up the training process.

Epoch

One complete pass through the entire training dataset. Multiple epochs are typically needed for a model to learn from the data.

Dropout

A regularization technique used in neural networks where a fraction of the neurons is randomly set to zero during training to prevent overfitting and improve generalization.

Optimizer

An algorithm used to adjust the model's parameters (weights) to minimize the loss function during training. Popular optimizers include Adam, SGD, and RMSprop, each with different strategies for updating weights.

Overfitting

When a model learns the training data too well, including noise, leading to high accuracy on the training set but poor performance on new data.

Distribution Shift

A change in the data distribution between training and testing or deployment, leading to decreased model performance because the model was not trained on data representative of the new conditions.