Exploring hierarchical label structure for better classification of drivers of tropical deforestation

by Fadel Mamar Seydou

Master's Thesis

Supervision

Prof. Devis Tuia, ECEO, EPFL
Thesis Advisor

Jan Pišl, ECEO, EPFL
Thesis Supervisor

EPFL ENAC IIE ECEO
CH-1951 Sion

July 19, 2023

Acknowledgments

I would like to express my sincere gratitude to the people who supported me throughout my journey. I would like to give my special thanks to Jan Pišl, who has greatly advised me during my thesis and helped me fit in the lab. It was a very challenging topic and benefiting from his knowledge has helped me grasp the content faster. I would also like to thank Prof. Devis Tuia for hosting me in his laboratory. His guidance has helped me a lot. My heartfelt thanks to my parents and friends for their support during my years of study in Switzerland. Thank you all.

Abstract

Tropical deforestation occurs at an alarming rate of approximately 5 million hectares of forest lost per year. This raises the need to develop accurate and timely quantification methods for drivers. In this thesis, we investigated tropical deforestation driver prediction as part of an image-level land-use and land-cover classification problem. Our key idea is to leverage the hierarchical structure of the annotations to improve the prediction performance on the leaf labels of the hierarchy. We explore hierarchical multi-label classification models that exploit this knowledge using hierarchy-aware architectures, hierarchy-aware loss functions, and hierarchy-aware pre-training. Our best model (F_1 score of 87.37%) outperforms our baselines by 0.45 percentage points on the leaf labels, and by 1.18 percentage points on the highest level of the hierarchy. We empirically show that hierarchy-aware pre-training can be useful in achieving better results. To access our code, please send an email to *fadel.seydou@gmail.com*.

Keywords: Deep Learning, Remote sensing, Land use and land cover classification, Hierarchical multi-label classification

Contents

Chapter 1

Introduction

Every year, the world loses approximately 5 million hectares of forest, up to 95% of which occurs in the tropics [23]. This is critical when we know that deforestation is the second-largest source of anthropogenic CO_2 emissions [6]. Although many drivers exist, the main drivers have been identified as pasture and agriculture. Effective mitigation actions require a better understanding of the deforestation trends that are currently performed through land use and land cover (LULC) mapping. Recently, with the increasing availability of free high-resolution satellite imagery, new methods based on machine learning have been proposed for producing more accurate LULC maps. The development of these new methods is justified by the need to automate the generation of LULC maps given the large quantity of free satellite imagery; Copernicus (the largest space data provider) delivers 16 terabytes of data every day.

The deep learning methods (e.g., U-Net segmentation model [25]) employed to generate LULC maps require a lot of manually annotated which can be difficult to access for the less dominant land uses (i.e. potential deforestation driver). Thus, the challenge is to develop machine learning models that can generalize well in the context of imbalances among the classes of interest. A particularity of LULC classification is the inherent relationship between target classes. They can be viewed as leaf nodes of a hierarchical graph built from their relationships.

To address the LULC classification problem, the concept of hierarchical labels was explored in this study. The key idea is to leverage the hierarchical structure of the annotations (labels) to produce more accurate maps. Typically, hierarchical graphs of annotations have fewer, more general classes at higher levels, and a larger number of more specific labels at lower levels. Research has suggested that such a structure may allow encoding of the differences between general classes at a higher level, leaving less information to be encoded at lower levels, thus requiring less data with specific labels[11, 13, 14, 19, 30]. As a study area, we selected Brazil, where 33% of the world's tropical deforestation occurs [23].

The goal of this thesis is to use the hierarchical structure of annotations to develop machine learning models that can generalize well in the setting of scarcity of annotated data for certain classes. Moreover, given that we are interested in drivers of deforestation and trends, we relaxed relax the pixel-level classification to an image-level classification. In summary, the **scope of this thesis** was

- Implement LULC classification models at the image level,
- Investigate how the inherent hierarchical structure of the annotations can be used to improve the prediction performance.

Chapter 2

Related Work

2.1 Tropical deforestation driver mapping

Tropical deforestation driver mapping has been studied from various perspectives, such as computer vision [1, 12, 30], landscape conservation [23], econometrics [21] and carbon emission mapping [6]. These studies cover methods ranging from simple questionnaires to more quantitative approaches using remote sensing and deep learning.

For instance, [9] assessed the presence of deforestation drivers across 28 tropical conservation landscapes around the tropics using questionnaires sent to landscape managers. The results provided were based entirely on expert knowledge and manual measurements; no machine learning was involved. In contrast, [21] attempted to identify the drivers and spatial patterns of deforestation in Colombia by using Landsat satellite imagery and econometric models. They found spatial autocorrelation between deforestation and population density, with dairy farming being the main driver. [6] also leveraged quantitative methods based on satellite imagery to connect post-deforestation land use to its corresponding carbon emission.

From a computer vision perspective, deforestation drivers were addressed as part of a LULC classification task, framed as a semantic segmentation problem. [1, 12, 30] addressed the LULC classification using satellite imagery and deep-learning classifiers. In our literature review, we focused on computer vision-related publications that were closer to the goal of our thesis.

2.2 Hierarchical multi-label classification

In a multi-label classification task, a given input can be associated with many labels simultaneously, which makes it more complicated than a multi-class classification task where a given input has only one class. In hierarchical multi-label classification (HMC), additional information on the relationships between the labels is available and can be leveraged. The relationship between labels can be described using a directed acyclic graph, where certain classes can be grouped into super-classes. Two main categories of methods have been developed to address this task [2, 4, 11]: **local methods** and **global methods**.

2.2.1 Overview of local and global methods

Local methods

In literature, there are three types of local methods (figure 2.1): *local per node*, *local per parent node* and *local per level*.

First, ***local per node*** methods build a classifier for each node of the hierarchy, excluding the root, and appear to be by far the most used in the literature [2]. These types of methods typically result in the development of many models that must later be associated using heuristics to make hierarchy-coherent predictions [4]. These methods require one to decide the design of a specific training dataset for each node in the hierarchy. [2] presents two ways to build such training datasets, namely the *inclusive* policy and the *sibling* policy. The inclusive policy label sub-classes are positive examples, and all others are negative examples, except for direct ancestors that are left out. The sibling policy differs from the inclusive policy by defining a negative example that comes only from sibling nodes. Second, ***Local per parent node*** methods train a classifier for each internal class to distinguish between its sub-classes. Both *local per parent node* and *local per node* scale poorly with the depth of the hierarchical graph and are only compatible with small classifiers that can be easily trained (e.g., random forests, logistic regression, and support vector machines) [22] which is not the case for deep neural networks. Finally, we have ***Local per level*** methods that train a multi-class classifier at each level of the hierarchy. Among the local methods, this set of methods is the most suitable for use along with deep neural networks.

Many publications [2, 4, 8, 22, 27] reviewed local methods in the context of gene and protein function prediction problems, where predictive trees or multi-layer perceptrons were used as classifiers. Publications related to our computer vision task [11, 13, 14, 19, 30] presented systematically global methods that used deep neural networks as classifiers. This observation led us to focus our reading on global methods.

Global methods

This approach is more attractive for deep-learning-based methods as a single model is developed to solve the HMC task (figure. 2.1), and interesting results have been reported. Some studies (e.g.[8] and [13]) have focused on crafting new hierarchy-aware loss functions that are very flexible and

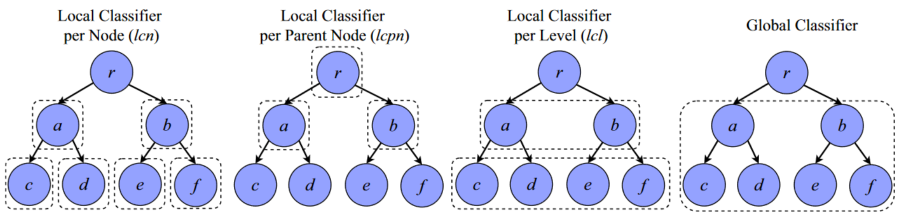

Local Classifier per Node (lcn)
Local Classifier per Parent Node (lcpn)
Local Classifier per Level (lcl)
Global Classifier
r
a
b
c
d
e
f

Figure 2.1: Local methods and global methods [24]

leverage knowledge from the hierarchy of classes. These objective functions are agnostic to the neural network architecture and use minimum and maximum operations to enforce hierarchy coherent predictions.

Other studies [11, 19, 30] designed hierarchy-aware neural network architectures that use multiple classification heads designed to predict classes at different levels of the hierarchy while sharing the same embeddings. They aimed to utilize both local and global information from the label hierarchy.

Another set of approaches [5, 13] focused on learning better representations (i.e., *hierarchical embeddings*) through contrastive learning where the goal is to learn embeddings whose distances (Euclidean or geodesic) mimicked the structure of the label hierarchy in a space R^d. For instance, [13] explored this path using a customized triplet loss, where the margin was defined as a function of the distance in the tree (i.e., the shortest path length) between the anchor and the positive and negative samples. On the other hand, [5] embedded both the label hierarchy and the inputs in a Poincaré hyperbolic space using a geodesic distance, which is stated to be more appropriate for encoding hierarchical structures.

Following the conclusion in [27] that "global models (predicting the complete structure as a whole) generally have better predictive performance than the local models" and the advantage with respect to scalability, we decided to focus on these models.

Chapter 3

Background and Design

This chapter will start by formulating the problem, then will present how the dataset was created.

3.1 Problem formulation

A robust classification model will help figure out the land uses (i.e., drivers) that induce deforestation over the years. We extracted annotations from the *MapBiomas Collection 6 dataset* of 2018 and used Sentinel-2 satellite imagery of 2018 as input source. In contrast to other works [1, 30] which address LULC classification as a semantic segmentation (i.e., pixel-level classification), we relaxed it to an image-level classification where the pixel-level majority class (superior to 60%) becomes the target. We believe that this relaxation decreases the influence of errors present at the pixel-level annotations explained in Section 3.2.

We frame our LULC classification task as a hierarchical multi-label classification problem where the goal is to use the hierarchy of labels to improve the prediction on the "leaf" labels (i.e. leaf nodes of the hierarchical graph of labels).

3.2 Source of data

3.2.1 MapBiomas Collection 6 dataset

The *MapBiomas Collection 6 dataset* is a collection of LULC maps of Brazil spanning the years 1985 to 2020, which are freely available and can be downloaded from the website of the MapBiomas project. The project, launched in 2015, created LULC maps at a resolution of 30 m per pixel using Landsat-8 satellite imagery [15]. These maps were obtained by classifying the pixels using random forest

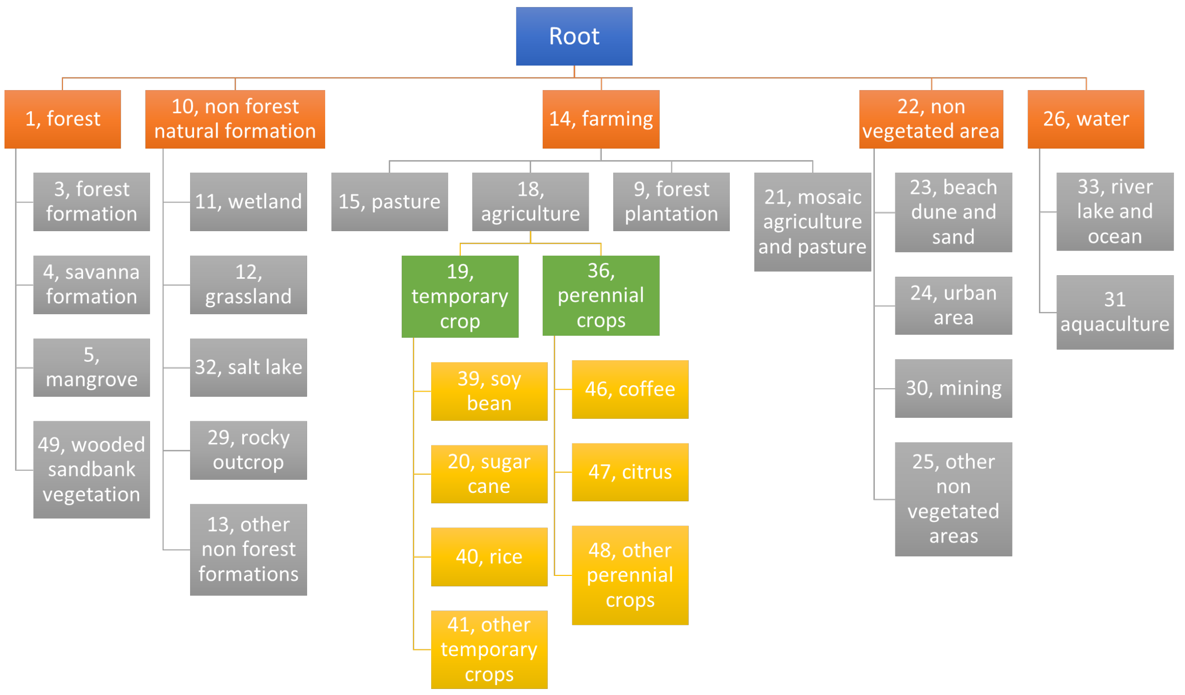


Figure 3.1: LULC from MapBiomass collection 6 [15]. The label hierarchy has 4 levels highlighted by different colors.

(tree-based method) and U-Net (deep learning-based method) [25]. The U-Net based classification model was applied to identify the classes "Aquaculture", "Irrigation", "Mining", "Rice" and "Citrus" while random forest was used for the remaining classes. No reason has been provided to support this choice. The semantic segmentation (i.e., pixel classification) models used to create the LULC maps showed a global accuracy between 87% and 91% and area disagreements between 2% and 3%. In light of these results, we selected the MapBiomas collection as a usable source of annotations (figure 3.1).

3.2.2 SENTINEL-2 satellite imagery

The Copernicus SENTINEL-2 mission comprises a constellation of two satellites monitoring the land surface conditions of Earth. It offers free high-spatial-resolution satellite imagery that is useful for LULC mapping. In our case, we accessed the 10 meter spatial resolution bands of the multi-spectral data: B2 (490 nm), B3 (560 nm), B4 (664 nm), and B8(842 nm) [29]. In the RBG format, B2 is blue, B3 is green, B4 is red, and B8 is the near infrared (NIR). We included band B8 (i.e., NIR) because it is very useful identifying vegetation. The normalized difference vegetation index (NDVI) used to identify vegetation relies on red and NIR bands [7]. SENTINEL-2 imagery was downloaded directly from the Google Earth Engine program using a dedicated script.

To allow an easier processing of the data with a convolutional neural network, the whole map of Brazil was discretized in blocks of $1km^2$ for which we only downloaded the satellite images of interest resized from 100 × 100 pixels to 256 × 256 pixels.

3.3 Data preparation

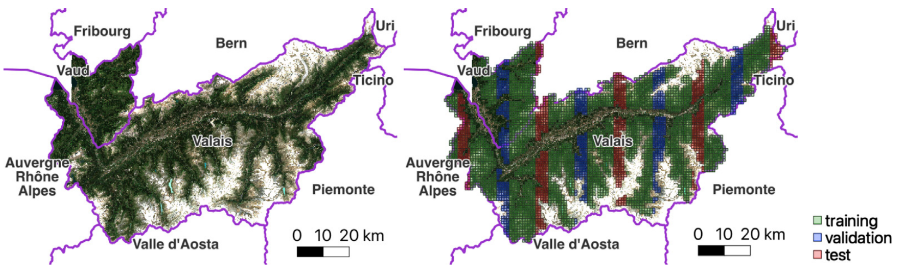
A very important part of this thesis was dedicated to partitioning the bulk of the data into training, validation, and test sets. The spatial auto-correlation between these sets, if not considered, violates the assumption that the data are identically and independently distributed. The partitioning has to be done diligently to avoid or decrease the spatial auto-correlation, which would result in leakage (i.e., knowledge of test samples available at training time) and consequently inflate the evaluation results. We would obtain over-optimistic models [26]. Indeed, if the test samples are very close to the training samples in space, we will not be able to assess the generalization of the model. Various methods exist in the literature to mitigate and decrease the risk of leakage. For instance, [26] used variograms (used in geostatistics to quantify spatial dependence) to create train-test splits of similar difficulty and assumed that real-world use locations are known. In contrast, [18] uses a simpler method that splits the area into large interleaving vertical strips with a width of 10 km for training and a width of 5 km for validation and testing. It uses the repeating schema detailed in figure 3.2 which we will take as inspiration to devise our splitting strategy. The spatial distribution of our classes (figure 3.3) is uneven and requires a different approach, especially because of the scale.

In this section we will cover the splitting of the data in training, validation and test sets.Here is the terminology used to describe our data:

- *block*: a square plot of land of area $1km^2$. The whole study area is divided into blocks with associated land use and land cover classes.
- *tile*: a square area of $25km^2$ constituted by 25 blocks.

3.3.1 Data partitioning

The setting in [18] is very different from ours for various reasons and made their method unsuitable. First of all, our study area which the whole of Brazil is more than 1400 times bigger (i.e. $8'515'767km^2$ instead of $5'897km^2$). Secondly, the topographies of the study areas and the distribution of the classes are very different. Our dataset consists of 25 different classes covering both LULC whereas [18] only looked at 4 very similar classes of forest (i.e. land cover) in a valley. As a result, we devised a partitioning strategy employing interleaving blocks that can be conceptualized by a matrix. The goal is to have all classes present in the three sets (i.e. training, validation, test) and ideally uniformly



(a) Sentinel-2 imagery over the study area (b) Image tiles repartition

Figure 3.2: Partitioning strategy of the study area (Canton of Valais in Switzerland) used as an inspiration[18].

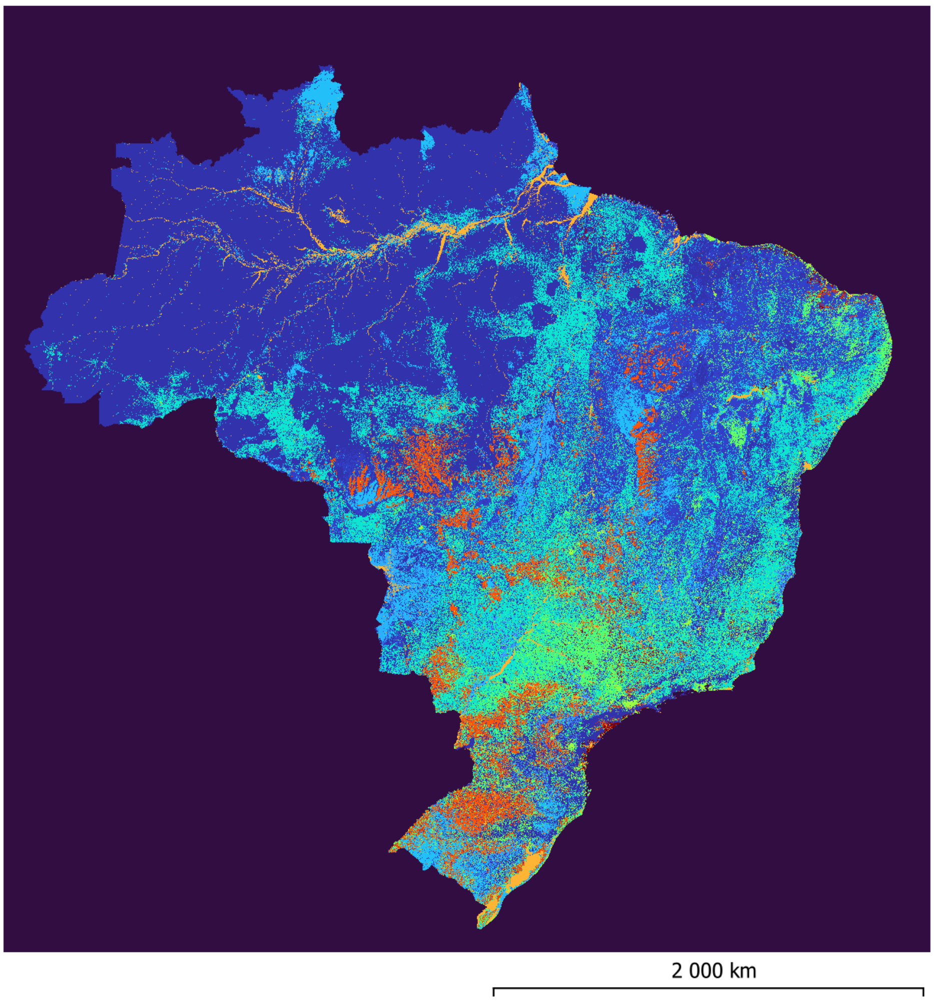


Figure 3.3: Spatial distribution of LULC classes from *MapBiomas Collection 6* of the year 2018 where every color corresponds to a specific class [15].

distributed. The pattern adopted to split the data is conceptualized by the matrix P described in equation 3.1 where the belonging to a partition for a block is given by $P[i, j]$.

$$
\begin{gathered}
P[i,j] = \begin{bmatrix} 0 & 2 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 2 & 1 \end{bmatrix} \\
i = \lfloor y/5 \rfloor \quad mod \quad 4 \\
j = \lfloor x/5 \rfloor \quad mod \quad 5
\end{gathered}
\tag{3.1}
$$

where 0, 1, 2 are tiles of the train set, validation set and the test set respectively; and x, y are the indices of a block.

The partitioning pattern has a band-like structure around the diagonal to separate the training, test, and validation sets and decrease the spatial auto-correlation. We iterated with different patterns and found that this provided a good distribution of the classes among the different sets. Additionally, with this pattern (figure 3.4), we can split the entire data such that approximately 60% belong to the training set, 20% belong to the validation set, and 20% belong to the test set.

Upon the application of the splitting strategy to the entire dataset, we divided the LULC maps into three non-overlapping sets for training, validation, and testing. Not all classes are evenly represented (figure 3.5), which is expected, but implies a need for a sampling of the data to have a balanced distribution of classes across these three sets.

3.3.2 Weighted Sampling of images

To further decrease the influence of spatial auto-correlations between the training, validation, and test sets, we used a weighted sampling approach such that blocks were sampled depending on their location within a tile. We sampled blocks that were further away from the boundaries of tiles with higher probability to increase the spatial distance between blocks from different sets. Weighting was computed using the probability density function of a normal distribution centered on the tile.

The sampled data represent the actual data used to train, validate, and test the classification models because using the entire dataset would dramatically increase the training time. When possible, we sampled 150 samples per class for the validation and test sets, and forced the training set to be three times larger (i.e., 450 samples per class). The normalized distribution of the LULC labels (figure 3.6) shows that the three sets are comparable in terms of the proportions of samples per class for most labels. We kept all the classes except class 32 (i.e., salt lake), which had only two images in the test set and ten images in the training set. We obtained the **training set** comprising

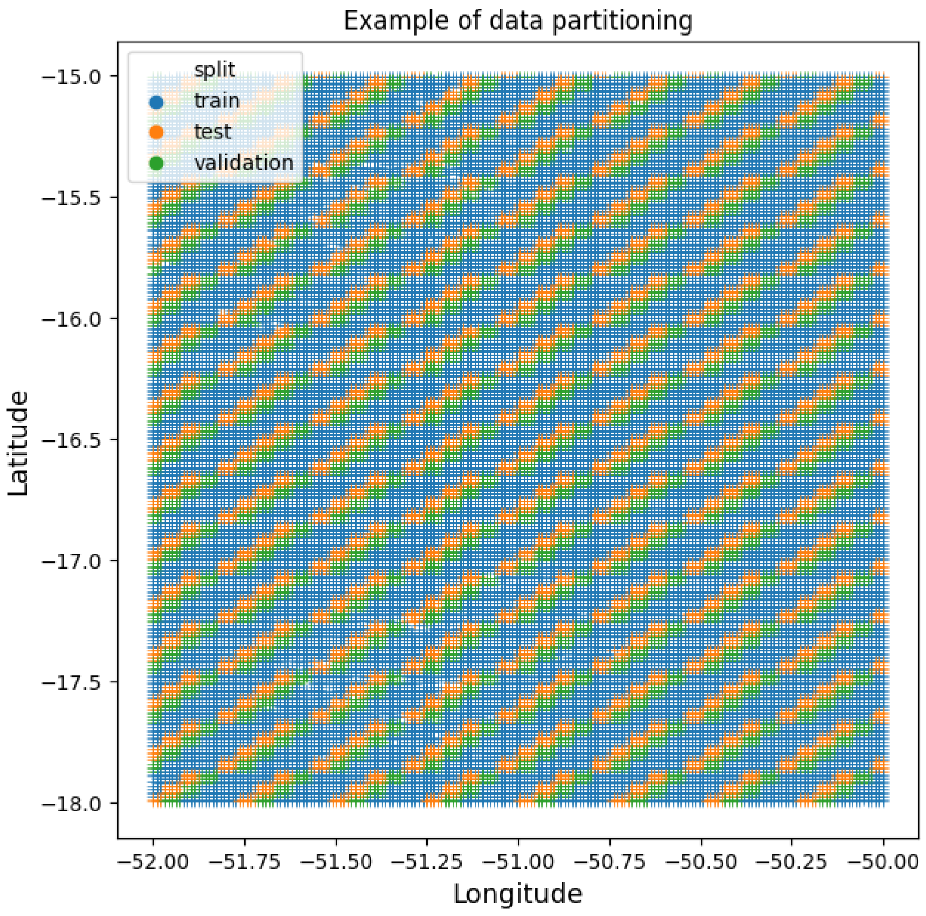


Figure 3.4: A visualization of the data partitioning strategy applied to a portion of our study area.

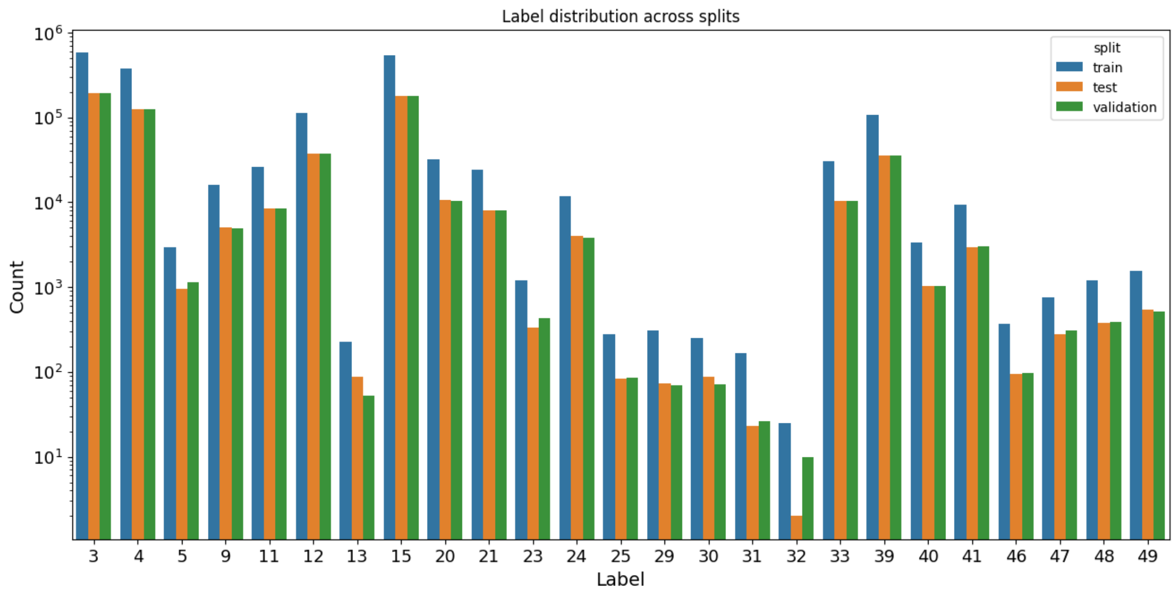


Figure 3.5: LULC distribution of the whole dataset. Note that only the leaf labels of the hierarchy are visible here.

9700 samples (60%), a **validation set** comprising 3102 samples (20%), and a **test set** comprising 3148 samples (20%). Figure A.1 presents the spatial distribution of the downloaded data. It is visible that the whole study area was covered, with the Amazon being less sampled than other areas as expected.

Qualitative assessment of the data

The quality of the downloaded data was visually assessed. Figure 3.7 shows satellite images from the training set. It is visible that some images or classes are quite different (e.g. figure 3.7f and figure 3.7e) while some can be quite close (e.g. figure 3.7a and figure 3.7b) which can be expected to have an influence on the false positive and false negative rates. Additionally, the differences in luminosity across the images suggest that the dataset might be difficult to handle, as some images might fall out of the distribution.

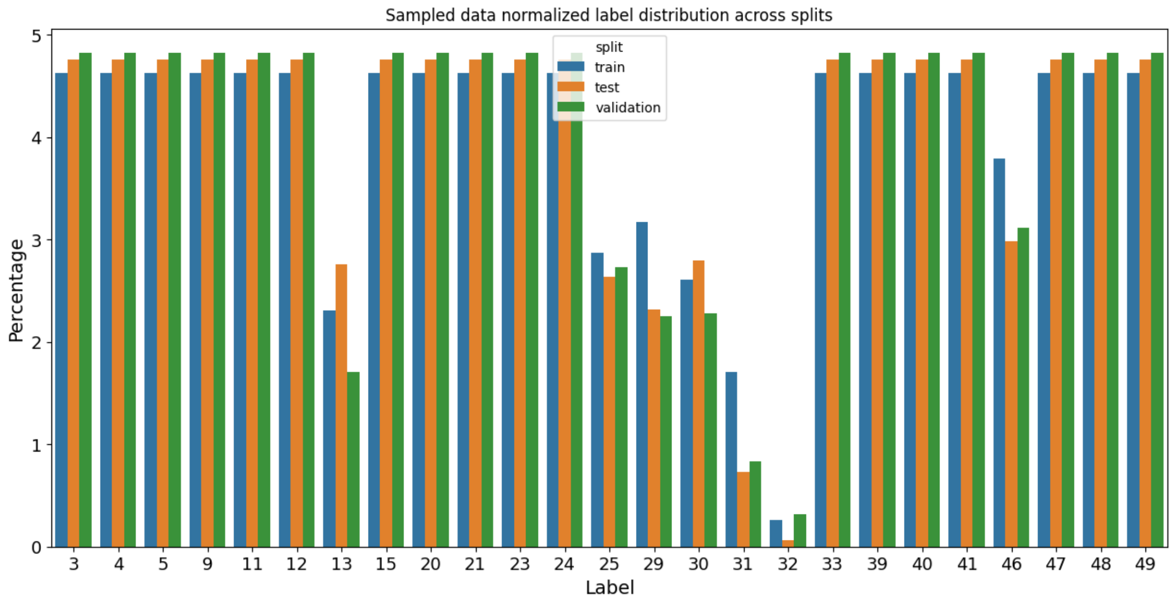
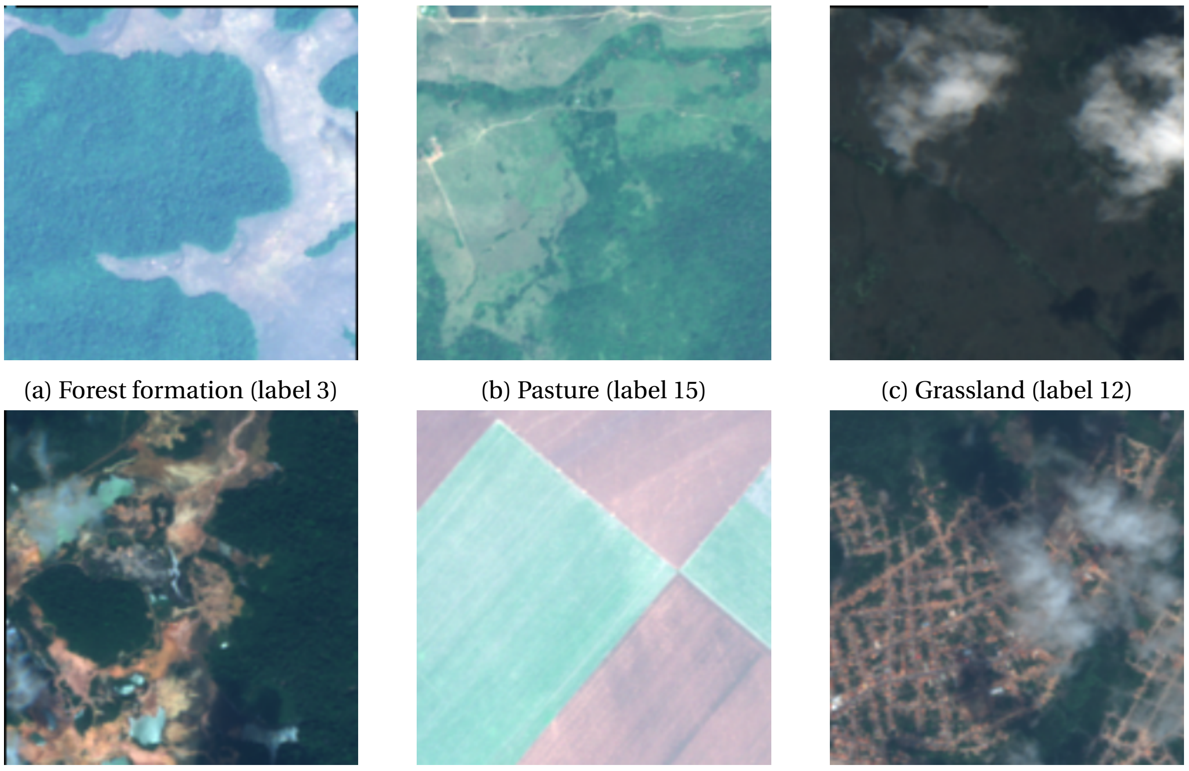


Figure 3.6: Normalized LULC distribution of the sampled dataset. The sampled dataset was used for the model training and testing.



(a) Forest formation (label 3) (b) Pasture (label 15) (c) Grassland (label 12)

(d) Mining (Label 30) (e) Soy bean (label 39) (f) Urban area (label 24)

Figure 3.7: Examples of input images from the training set. Every image represents an area of $1km^2$.

Chapter 4

Methods

This chapter will cover the different methods implemented in the thesis. The baseline classification model will be presented first, then followed by the hierarchical multi-label classification models.

4.1 Baseline classification model

To assess the potential improvements brought in by the hierarchical multi-label classification models, we have developed three baseline classification models which ignore the hierarchy and classify the leaf labels, level 1 and level 2 labels of the hierarchy (figure 3.1). Level 3 and 4 labels are not considered due to the data being insufficient.

This method is straightforward; we solve **multi-class classification** tasks on the leaf labels, on the labels at level 1, and the labels at level 2 by minimizing the cross-entropy loss function defined as:

$$\mathcal{L}^{baseline} = -\sum_{i=1}^{N} y_i \times ln(h_i) \tag{4.1}$$

where h_i is the output probabilities for label i, y_i is the target probability for label i and N is the number of labels considered.

4.2 Hierarchical multi-label classification models

4.2.1 Vanilla multi-label classification

The vanilla multi-label classification model does not explicitly leverage knowledge from the hierarchy, but rather implicitly through the co-occurrences of labels. This allowed us to assess whether any performance increase was induced by explicitly leveraging prior knowledge from the label hierarchy. In this case, the predictions can still violate the hierarchy as no constraints are set. The model was trained using a binary cross-entropy (BCE) loss function (equation 4.2), where both parent and leaf labels were available. All classes are predicted except for class 32 which was discarded, as explained in Section 3.3.2.

For a given input, the BCE loss is given by:

$$\mathscr{L}^{BCE} = -\sum_{i=1}^{33} y_i \times ln(h_i) + (1 - y_i) \times ln(1 - h_i) \tag{4.2}$$

where h_i is the output probabilities for label i, y_i is the target probability for label i.

4.2.2 Hierarchy-aware loss functions

The first set of global methods implemented to address the hierarchical multi-label classification problem are the hierarchy-aware loss functions. The *MCLoss* function (equation 4.6) was originally employed for gene function prediction while the*Tree-min* loss function (equation 4.10) was introduced for semantic segmentation. We adapted both to our use case of multi-label image classification.

MCLoss

The MCLoss or *max constraint loss* [8] is a binary cross-entropy (BCE) loss with modified inputs designed to fulfill the hierarchy constraint. The MCLoss relies on a *maximum constraint module* (MCM) that enforces the hierarchy constraint given some output probabilities. The MCM is appended to a base classifier and enforces that the probability of a parent class is the maximum probability among its sub-classes. It implies that the prediction of a class can be delegated to its sub-classes enforcing coherence with the hierarchy.

For any class A in the set S of hierarchically structured classes, the *maximum constraint* is defined as:

$$MCM(h_A) = max_{B \in D_A} h_B \tag{4.3}$$

where $h_A \in [0,1]$ are the predicted probabilities for class A by the base classifier, D_A is the set of sub-classes of A. Any class A belongs to its own set of sub-classes.

It follows that for every class A , $MCLoss_A$ is defined as:

$$\mathscr{L}_A^{MC} = -y_A \times ln(\tilde{h}_A) - (1 - y_A) \times ln(1 - MCM_A) \tag{4.4}$$

where y_A is the ground-truth label for class A, $MCM_A = MCM(h_A)$ and

$$\tilde{h}_A = max_{B \in D_A}(y_B \times h_B) \tag{4.5}$$

The final MCLoss is defined by:

$$\mathscr{L}^{MC} = \sum_{A \in S} \mathscr{L}_A^{MC} \tag{4.6}$$

At the inference stage, the hierarchically coherent multi-label prediction vector h is given by:

$$h = [MCM(h_{A_1})...MCM(h_{A_n})] \tag{4.7}$$

where h_{A_i} is the output probability of the base classifier for class A_i with $i \in [1, n]$.

Tree-Min loss

The Tree-min loss [13] is very similar to the MCLoss in the sense that similar constraint modules are used and it also boils down to is a binary cross-entropy loss with modified inputs. The differences lie in the computation of the output probabilities at training time and inference time. At training time, the hierarchy constraints are enforced using two properties; the *negative property* and the *positive property*. The positive property states that if a class is labeled positive, all its ancestor classes (i.e. super-classes) should be labeled positive. The negative property states that if a class is labeled negative then all its descendant classes (i.e. sub-classes) should be labeled negative. It is to be noted that any class A if considered to its own ancestor and its own descendant. The positive property is translated into a *minimum constraint module* (MinCM) and the negative property is translated into a *maximum constraint module* (MCM i.e. identical to the one presented in section 4.2.2).

For any class A in the set S of hierarchically structured classes, the minimum constraint is defined as:

$$MinCM(h_A) = min_{B \in U_A} h_B \tag{4.8}$$

where U_A is the set of super-classes of A and h_A are the predicted probabilities for class A by the base classifier.

It follows that for every class A, the loss is defined as:

$$\mathscr{L}_A^{TM} = -y_A \times ln(s_A)) - (1 - y_A) \times ln(1 - MCM(h_A)) \tag{4.9}$$

where y_A is the ground-truth label for class A, $s_A = MinCM(h_A)$ and $MCM(h_A)$ follows equation 4.3. The minimum constraint is applied to the positive labels while the maximum constraint is applied to the negative labels.

The final Tree-Min loss is given by:

$$\mathscr{L}^{TM} = \sum_{A \in S} \mathscr{L}_A^{TM} \tag{4.10}$$

In the inference stage, each input image is associated with the top scoring root-to-leaf path P^* in the class hierarchy H as follows:

$$P^* = argmax_{P \subseteq H} \sum_{u \in P} h_u \tag{4.11}$$

4.2.3 Hierarchy-aware architectures

The second set of global methods implemented, focused on adapting the classification heads of the neural networks so as to mimic the structure of the hierarchical structure [4, 11, 14, 19, 22, 30], hence *hierarchy-aware architectures.* The idea is to solve a multi-class classification task at each level using dedicated classification heads. In opposition to methods using hierarchy-aware loss functions, these architectures do not have a post-processing module enforcing the coherence with the hierarchy. It is assumed that the relationship between the levels is learned. Among these customized neural network architectures, we paid a special attention to [4, 11] due to their intuitive formulation.

The first hierarchy-aware neural network architecture (named **custom-Arch 1**) investigated was inspired by [11]. It is composed by an image encoder and 4 classification heads. Each classification head l (i.e. linear layer) takes in charge the prediction of its associated level. The general architecture is depicted in figure 4.1 with the output of head $l-1$ flowing into head l without applying any activation function. The input z_l to classification head l is defined:

$$z_l = x_{l-1} \tag{4.12}$$

where x_{l-1} is the output of head $l-1$.

In [11], a common image encoder (ResNet18) is trained jointly with the linear layer of each level of the hierarchy graph (figure 4.1). In essence, it bridges the gap between *local per level* and global models by allowing the model to access both global information and local information thanks to the

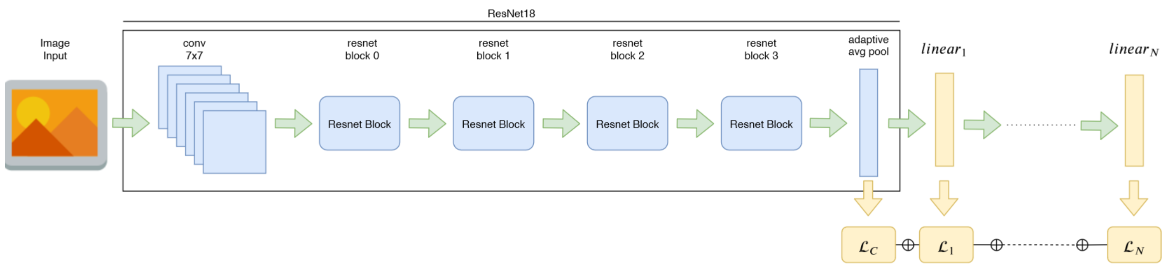


Figure 4.1: An example of hierarchy-aware neural network architecture [11]

structure of the loss function and the back-propagation. For a given input, and label hierarchy of N levels, the loss is defined as:

$$\mathscr{L}^1_{HA} = \|x - c_y\|^2_2 + \sum_{l=1}^{N} CE(h_l, y_l) \tag{4.13}$$

where h_l is the output probabilities from layer l, y_l is the target at level l, x is the embedding of the input image, c_y is the center of the class y in the feature space of the deep model and

$$CE(h_l, y_l) = -\sum_{i=1}^{m_l} y_{l_i} \times ln(h_{l_i}), \quad \text{with} \quad y_l \in [0,1]^{m_l}, h_l \in (0,1]^{m_l} \tag{4.14}$$

The second hierarchy-aware neural network architecture (named **custom-Arch 2**) investigated was inspired by [4]. It is comprised by an image encoder and 4 classification heads designed similarly to figure 4.2. The difference with custom-Arch 1 is that each classification head l has also access to the embedding learnt by the image encoder. The input z_l to classification head l is defined as:

$$z_l = CONCAT([x, x_{l-1}]), \quad l \in 2,3,4 \quad z_1 = x \tag{4.15}$$

where $CONCAT$ is the concatenation operation and $x \in R^d$ is the image embedding and x_{l-1} is the output of head $l-1$.The loss function is defined as:

$$\mathscr{L}^2_{HA} = \sum_{l=1}^{N} CE(h_l, y_l) \tag{4.16}$$

where h_l is the output probabilities from layer l, y_l is the target at level l and $CE(h_l, y_l)$ follows equation 4.14.

NB: many classes do not have descendants at level 3 and 4, so dummy labels at those levels acting like background labels were added. This yielded better results compared to simply discarding inputs images without a class at level 3 and 4.

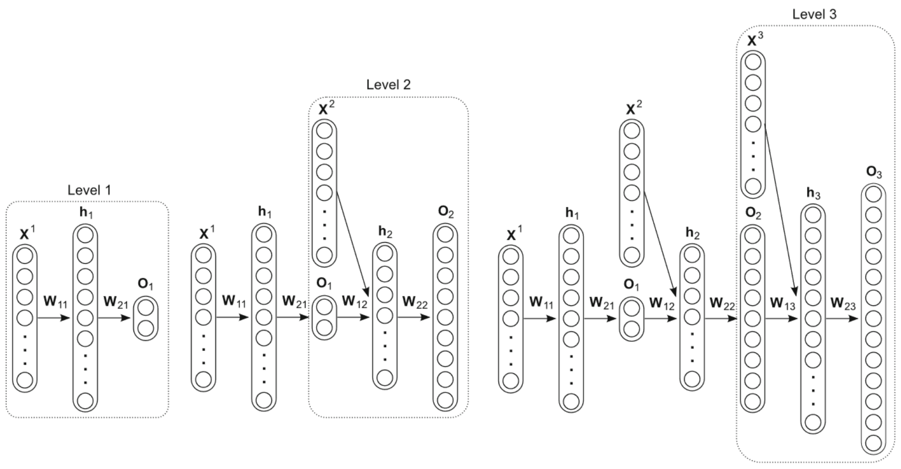


Figure 4.2: An example of hierarchy-aware neural network architecture [4]

4.2.4 Hierarchy-aware semi-supervised pre-training

An interesting idea developed in [13] revolved around improvement the representation power of the image encoder. They claimed that in there case of hierarchical semantic segmentation (i.e. pixel level classification), they had better results by improving the representation power of the image encoder using contrastive learning. The hierarchy-aware semi-supervised learning is a contrastive learning approach with the goal of bringing similar instances closer in the representation space. In this hierarchical classification context, instances that belong to classes closer in the hierarchy graph (with respect to the length of the shortest path) are expected to become closer in the representation space.

Inspired by [13], we implemented the *Tree-triplet loss* ($\mathscr{L}^{TT}$ in equation 4.17) and used it in two ways. First, it was used as part of the combinatorial loss $\mathscr{L}^{CM}$ (equation 4.19) and then it was used to pre-train the image encoder. The hierarchy-aware triplet-loss function (i.e., *Tree-triplet loss*) is minimized and has the following formulation:

$$\mathscr{L}^{TT} = max\{d(i^+, i) - d(i^-, i) + m, 0\} \tag{4.17}$$

where $d(a, b) = 0.5 * (1 - \frac{a \cdot b}{\|a\| \|b\|})$ is the cosine distance , i^+ is the embedding of the positive example, i^- is the embedding of the negative example and i is the embedding of the anchor.

The separation margin m which forces the gap of $d(i^+, i)$ and $d(i^-, i)$ to be larger than m is defined as:

$$m = 0.1 + 0.5 \times (\phi(i^-, i) - \phi(i^+, i))/(2 * D) \tag{4.18}$$

where D is the height of the hierarchy graph (i.e. 4 in our case) and $\phi(a, b)$ is the number of edges on the shortest path linking a and b in the label hierarchy.

Additionally, a combinatorial loss $\mathscr{L}^{CM}$ (equation 4.19) leveraging both semi-supervised learning and hierarchy-aware loss function was defined as:

$$\mathscr{L}^{CM} = \mathscr{L}^{TM} + \beta \times \mathscr{L}^{TT} \tag{4.19}$$

where $\mathscr{L}^{TM}$ follows equation 4.10 and $\beta \in [0, 0.5]$ is scheduled using a cosine annealing policy [13].

4.3 Experimental setup

4.3.1 Training details

The actual implementation of the neural network and the training routine were done in pyTorch [20] using one GPU NVIDIA GeForce GTX TITAN X. For all the models, we use Adam [10] as the optimizer with hyperparameters: $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 10^{-8}$ and a tuned weight decay of 10^{-5}. The base learning rate was 10^{-4} for all models except *custom-Arch 1* for which it was 10^{-3}. Learning rate was scheduled using a multiplicative scheduler [20] of factor 0.85 during the first 15 epochs and then it was kept constant. This scheduling showed to limit over-fitting. The batch size was set to 32 due to the limitation of GPU memory.

For a fair comparison, all models were trained for 15 epochs with the same EfficientNet-b3 [28] pretrained on ImageNet-1K [20] as the image encoder. The maximum number of epochs was set to 15 after noticing a stagnation of validation metrics after 10 epochs (figure A.3). It also reduces energy consumption. The best weights (which achevied the lower validation loss) were saved for evaluation. EfficientNet-b3 performed better than a ResNet-18 justifying its choice. The embedding dimension of the image encoder is 384.

4.3.2 Data augmentation

The data augmentation applied here was standard. At training time, we applied with probability $p = 0.5$, a *Shift-Scale-Rotate* transformation with rotation limit 180 degrees, a *Horizontal Flip* transformation, a *Gaussian Blur* transformation with kernel size $k \in [3, 7]$, and finally we normalized the data. All transformation were implemented using Albumentations [3] and torchvision [20]. At inference time, the input data was only normalized. Note that the input normalization statistics were computed on the training set.

4.3.3 Evaluation

We remind that although we trained hierarchical multi-label classification models, we are mainly interested in the improvements (if any) on the leaf labels of the hierarchy; meaning that the evaluation will follow that of a multi-class classification on the leaf labels. To get clues about potential failure modes of the hierarchical multi-label classification models, we computed the prediction performances on the labels at level 1 and 2.

The evaluation metric selected is the F_1 score (equation 4.20) which is more suited to our setting of imbalance distribution. This evaluation metric summarizes both the precision and the recall by computing their harmonic mean. We use macro-averaging (i.e. computing the score for each class and then averaging it) to attribute the same weight to leaf label. Additionally, we compute the confusion matrix to understand which ones are being confounded.

$$F_1 = \frac{2 \times TP}{2 \times TP + FP + FN} \tag{4.20}$$

where TP is the number of true positives, FP is the number of false positives, FN is the number of false negatives and TN is the number of true negatives.

Chapter 5

Results and Discussion

This chapter will presents and discuss the results of the experiments.

5.1 Results on leaf labels

5.1.1 Quantitative results

The column "F_1 score (leaf)" of Table 5.1 shows the macro-averaged F_1 scores on the leaf labels.

The baseline model trained to solve a multi-class classification problem on the leaf labels is not (significantly) outperformed by any of the hierarchical multi-label models implemented. The best hierarchical multi-label model (results in bold in table 5.1) outperforms the baseline by only 0.45 percentage points. The other hierarchical multi-label models are roughly similar (around 86%) except:

- those associated with "Tree-min" and "Tree-min+Tree-triplet loss functions. These two hierarchy-aware loss functions [13] seem to not work well for our dataset. In conjunction with the base network, the performance decreases by non negligible 2.83 and 2.31 percentage points respectively; while with the *custom-Arch 2* network, the performance decreases by 1.16 and 1.56 percentage points respectively.
- those associated with *custom-Arch 1* which were much worse (between 43.14 and 16.26 percentage points lower). This hierarchy-aware architecture assumes that the prediction at level $l-1$ can be used to make predictions at level l as explained by equation 4.12. It happens to work poorly compared to custom-Arch 2 which uses the embeddings from the image encoder and the previous prediction (equation 4.15).

On the leaf nodes, the best model was the one trained with both a hierarchy-aware architecture (*custom-Arch 2*) and a (MCLoss) hierarchy-aware loss function with a pre-trained image encoder. It achieved an F_1 score of 87.37% a little bit higher than the 86.92% of the baseline. However, the figure 5.1 shows that at the level on individual leaf labels, the trend is not unique. For example, the baseline works better for the "other temporary crops" and "soy bean" classes. On this set of labels, the results suggest that having a good inductive bias when designing new architectures can be helpful.

NB: although the results on the leaf labels seem quite high, they are consistent with the semantic segmentation performance reported by the models used to create our ground-truth annotations (section 3.2). Therefore, we do not suspect leakage between the train and test datasets.

Influence of semi-supervised pre-training

We tried a hierarchy-aware semi-supervised learning according to equation 4.17 in two ways; first as part of the combinatorial loss (Tree-min+Tree-triplet in table 5.1) and then secondly as to pre-train the image encoder. **The results were interesting** and encouraged towards investigating better pre-training methods.

- When using "Tree-triplet" loss with the "tree-min" loss, the results were generally lower than that of just training with with "tree-min" loss. Results decreased by 0.4, 0.98 percentage point with *custom-Arch 2* and *custom-Arch 1* respectively, while they increased by 0.52 with the base network.
- In the pre-training case, the results depended on whether the image encoder was frozen or not. We tried it only with the best performing model on the leaf nodes (i.e. *custom-Arch 2* with MCLoss). A sharp drop of 21.49 percentage points was noticed when the image encoder was frozen, while a slight increase of 0.41 was observed in the opposite case.

5.1.2 Qualitative results

From the confusion matrix (figure A.2) and results computed for each leaf label (figure 5.1), we can see that with some other classes ("coffee", "aquaculture", "urban area" and "river lake and ocean") were easily predicted, both baseline and the best hierarchical multi-label models struggle with the classes "mosaic agriculture and pasture" and "other temporary crops". This actually stems from the labelling of those classes. The manuscript describing the annotation procedure [16] states:

- "other temporary crops" characterizes crops with vegetative cycle of less than a year. It is confounded a lot with "soy bean" which are both sub-classes of "temporary crops". 21% of actual "other temporary crops" are predicted as "soy bean".

- "mosaic agriculture and pasture" characterises areas where it was not possible to distinguish between pasture and agriculture. Coincidentally, the best model confounds this class mainly with "pasture" (9%) and "other temporary crops" (9%) and "sugar cane" (7%).

We visualized the centroids' embeddings of each class in figure A.4 to get a sense of the improvement brought by the semi-supervised pre-training. Looking at the embeddings from the pre-trained encoder (figure A.4b), we see that "sibling" labels (in the hierarchy) and their descendants are closer than other labels. For example, descendants of "temporary crop" (labels 39, 20, 40 41) form a cluster located far away from descendants of "non vegetated area" (labels 23, 24,30,25) but close to descendants of "perennial crops" (labels 46, 47, 48). **After fine-tuning the pre-trained image encoder, more semantic similarities were depicted** (figure A.4a). For instance, "mangrove" (label 5) is brought closer to "river, lake and ocean" (label 33) and "wetland" (label 11).

Neural networks	Loss functions	F_1 score (leaf)	F_1 score (Level 1)	F_1 score (Level 2)
Base network (baselines)*	CE (Eq. 4.16)	86.92	93.82	86.98
Base network	BCE (Eq. 4.2)	86.58	94.01	86.89
Base network	MCLoss (Eq. 4.6)	85.55	94.36	86.09
Base network	Tree-min (Eq. 4.10)	84.09	93.06	84.96
Base network	Tree-min+Tree-triplet (Eq. 4.19)	84.61	93.91	85.22
custom-Arch 2	CE (Eq. 4.16)	86.42	94.03	87.01
custom-Arch 2	BCE (Eq. 4.2)	86.11	94.23	86.89
custom-Arch 2	MCLoss (Eq. 4.6)	86.96	94.77	87.75
custom-Arch 2	Tree-min (Eq. 4.10)	85.76	94.71	86.55
custom-Arch 2	Tree-min+Tree-triplet (Eq. 4.19)	85.36	94.45	86.40
custom-Arch 2 **	MCLoss (Eq. 4.6)	65.47	92.41	69.33
custom-Arch 2 ***	MCLoss (Eq. 4.6)	**87.37**	**94.87**	**88.16**
custom-Arch 1	CE (Eq. 4.16)	43.78	92.28	48.69
custom-Arch 1	BCE (Eq. 4.2)	67.04	94.75	84.31
custom-Arch 1	MCLoss (Eq. 4.6)	70.66	94.11	84.86
custom-Arch 1	Tree-min (Eq. 4.10)	62.21	94.03	81.36
custom-Arch 1	Tree-min+Tree-triplet (Eq. 4.19)	61.23	94.46	80.08

Table 5.1: Macro-averaged F_1 score (in percentage) on the test dataset for leaf labels, level 1 labels, and level 2 labels. "*" designates baseline models trained in a multi-class setting. "**" characterizes an image encoder pre-trained using Treetriplet loss (Eq. 4.17) but frozen at training. "***" characterizes an image encoder pre-trained using Treetriplet loss (Eq. 4.17) and finetuned at training. The "base network" is a neural network with an image encoder and one classification head. The highest values on each set of labels are in **bold**.

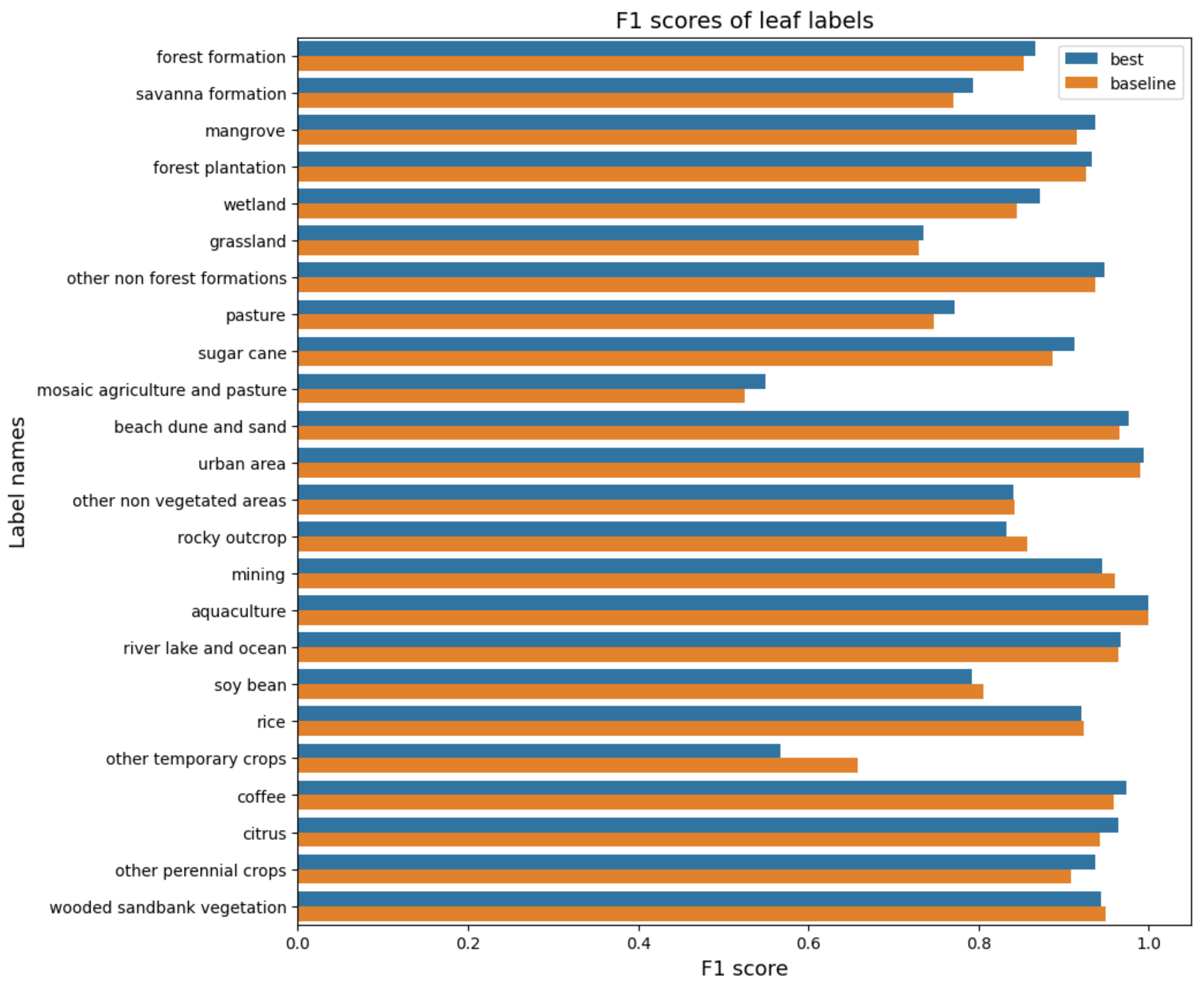


Figure 5.1: F_1 score computed for each leaf label."Best" denotes the *custom-Arch 2* trained with MCLoss.

5.2 Results at higher levels of the hierarchy

5.2.1 Quantitative results

The columns "F_1 score (Level 1)" and "F_1 score (Level 2" of Table 5.1 show the results on the level 1 and 2 labels respectively. The results are different depending on the level:

- At level 1 ("forest","non forest natural formation","farming","non vegetated area","water"), the results are quite high as expected. The baseline achieves an F_1 score of 93.82% and is outperformed by the our best model by 1.05 percentage points. The other hierarchical multi-label models except those associated with *custom-Arch 1* achieved either slightly lower or slightly higher results. The same observation is made on the importance of good inductive biases when designing new hierarchy-aware architectures.

- At level 2, the results are globally a bit higher than those on the leaf nodes but they are more or less similar. We remind that due to structure of the hierarchy, 17 labels out of the 24 leaf labels belong to level 2 labels (there is quite a big overlap between these two sets of labels). Our best model outperforms the baseline by only 1.18 percentage points.

In summary, the hierarchy-aware loss functions ("tree-min" and "MCLoss") help achieve better results on the labels at levels 1 and 2. The hierarchical multi-label models are unique global models, and they were compared to three baseline models trained at levels 1, 2, and on leaf labels. In a production environment, a unique performing model would require less maintenance and would be easier to use; this is the main competitive advantage of hierarchical multi-label models.

5.2.2 Qualitative results

Globally speaking, the classification at the level 1 seem much easier than on the leaf labels or at level 2. This result was expected as there are fewer classes at the top of the hierarchy and they tend to be very different (figure 3.7).

5.3 Limitations

Based on our results, we believe that the main limitation comes from the quality of the annotations. As explained in Section 3.2, the land use and land cover maps used to extract our ground-truth annotations were the by-products of other semantic segmentation models. We cannot perform any better than the actual models providing us with our ground truth. Although the segmentation models achieved good results, some uncertainties remained in the data. The manuscript describing the data [16] explained that the class "mosaic agriculture and pasture" could be either "pasture" or "agriculture'. Coincidentally, all models didn't successfully predict that class. Nevertheless, we strongly believe that further research combining hierarchy-aware loss functions and hierarchy-aware architectures can help achieve better results on better datasets.

Chapter 6

Conclusion

Tropical deforestation is happening at the alarming rate of around 5 million of hectares of forest lost per year. It raises the need for the development of accurate and timely quantification methods for its drivers. In this thesis, we have investigated tropical deforestation drivers prediction as part of an image-level land use and land cover classification problem. We sampled satellite imagery of 10-m resolution in Brazil using Google Earth Engine and associated it with land use and land cover maps from the mapBiomass project to produce our dataset.

Our main goal was to investigate how the hierarchical structure of land use and land cover labels could be used to improve the prediction performance on the leaf label of the hierarchy. This was motivated by the observation that some tropical drivers (land uses) do not have sufficient annotations but should still be accurately predicted. We investigated two types of hierarchical multi-label classification models, namely hierarchy-aware loss functions and hierarchy-aware architectures, and a hierarchy-aware pre-training method. On the leaf labels of our hierarchy graph, we achieved approximately the same performance as baseline model. However, for the labels at levels 1 and 2, we achieved slightly better results. In summary, the best performing hierarchical multi-label model achieved better results than our three baseline models.

Although we were not able to achieve results significantly better than our baseline, such as in [30], our experiments showed that one hierarchical multi-label model could replace three baseline models. Thus, our hierarchical multi-label model would be more suited to a production environment where managing multiple models could be cumbersome. We believe that further research with good inductive biases can help achieve better results using both hierarchy-aware loss functions and hierarchy-aware architectures.

Bibliography

[1] Kamran Ali and Brian A. Johnson. "Land-Use and Land-Cover Classification in Semi-Arid Areas from Medium-Resolution Remote-Sensing Imagery: A Deep Learning Approach". In: *Sensors* 22.22 (2022). ISSN: 1424-8220. DOI: 10.3390/s22228750.

[2] M. S. Bewley, N. Nourani-Vatani, D. Rao, B. Douillard, O. Pizarro, and S. B. Williams. "Hierarchical Classification in AUV Imagery". In: *Field and Service Robotics: Results of the 9th International Conference*. Springer Tracts in Advanced Robotics. Cham: Springer International Publishing, 2015, pp. 3–16. DOI: 10.1007/978-3-319-07488-7_1.

[3] Alexander V. Buslaev, Alex Parinov, Eugene Khvedchenya, Vladimir I. Iglovikov, and Alexandr A. Kalinin. "Albumentations: fast and flexible image augmentations". In: *CoRR* abs/1809.06839 (2018). arXiv: 1809.06839.

[4] Ricardo Cerri, Rodrigo C. Barros, André C. P. L. F. de Carvalho, and Yaochu Jin. "Reduction strategies for hierarchical multi-label classification in protein function prediction". In: *BMC Bioinformatics* 17.1 (Sept. 15, 2016), p. 373. ISSN: 1471-2105. DOI: 10.1186/s12859-016-1232-1.

[5] Boli Chen, Xin Huang, Lin Xiao, Zixin Cai, and Liping Jing. *Hyperbolic Interaction Model For Hierarchical Multi-Label Classification*. version: 2. Sept. 4, 2019. arXiv: 1905.10802[cs, stat].

[6] V De Sy, M. Herold, F. Achard, V Avitabile, A. Baccini, S. Carter, J. G. P. W. Clevers, E. Lindquist, Maria Pereira, and L. Verchot. "Tropical deforestation drivers and associated carbon emission factors derived from remote sensing data". In: *ENVIRONMENTAL RESEARCH LETTERS* (2019). DOI: 10.1088/1748-9326/ab3dc6.

[7] *GISGeography*.

[8] Eleonora Giunchiglia and Thomas Lukasiewicz. *Coherent Hierarchical Multi-Label Classification Networks*. version: 1. Oct. 20, 2020. arXiv: 2010.10151[cs,stat].

[9] H. Manjari Jayathilake, Graham W. Prescott, L. Roman Carrasco, Madhu Rao, and William S. Symes. "Drivers of deforestation and degradation for 28 tropical conservation landscapes". In: *AMBIO* (2021). DOI: 10.1007/s13280-020-01325-9.

[10] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[11] Riccardo La Grassa, Ignazio Gallo, and Nicola Landro. *Learn Class Hierarchy using Convolutional Neural Networks.* May 18, 2020. arXiv: 2005.08622[cs].

[12] *Land-cover classification with high-resolution remote sensing images using transferable deep models.* DOI: 10.1016/j.rse.2019.111322.

[13] Liulei Li, Tianfei Zhou, Wenguan Wang, Jianwu Li, and Yi Yang. *Deep Hierarchical Semantic Segmentation.* Mar. 29, 2022. arXiv: 2203.14335[cs]. (Visited on 03/22/2023).

[14] Xiaodan Liang, Eric Xing, and Hongfei Zhou. "Dynamic-Structured Semantic Propagation Network". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Salt Lake City, UT: IEEE, June 2018, pp. 752–761. DOI: 10.1109/CVPR.2018.00085.

[15] *MapBiomass Collections.*

[16] *MapBiomass General handbook.*

[17] Leland McInnes, John Healy, and James Melville. *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction.* 2020. arXiv: 1802.03426 [stat.ML].

[18] Thiên-Anh Nguyen, Benjamin Kellenberger, and Devis Tuia. "Mapping forest in the Swiss Alps treeline ecotone with explainable deep learning". In: *Remote Sensing of Environment* (2022). Publisher: Elsevier. DOI: 10.1016/j.rse.2022.113217.

[19] Khondaker Tasrif Noor, Antonio Robles-Kelly, and Brano Kusy. "A Capsule Network for Hierarchical Multi-label Image Classification". In: *Structural, Syntactic, and Statistical Pattern Recognition, S+sspr 2022.* Vol. 13813. ISSN: 0302-9743 WOS:000927580200017. Cham: Springer International Publishing Ag, 2022, pp. 163–172. DOI: 10.1007/978-3-031-23028-8_17.

[20] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Z. Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *CoRR* abs/1912.01703 (2019). arXiv: 1912.01703.

[21] Cristian D. Ramirez, Sergio A. Orrego, and Laura C. Schneider. "Identifying Drivers and Spatial Patterns of Deforestation in the Rio Grande Basin, Colombia". In: *JOURNAL OF LATIN AMERICAN GEOGRAPHY* (2018). DOI: 10.1353/lag.2018.0005.

[22] André C.P.L.F. de Carvalho Ricardo Cerri Rodrigo C. Barros. *Hierarchical multi-label classification using local neural networks.* DOI: 10.1016/j.jcss.2013.03.007.

[23] Hannah Ritchie and Max Roser. "Forests and Deforestation". In: *Our World in Data* (2021). https://ourworldindata.org/forests-and-deforestation.

[24] Miguel Romero, Oscar Ramírez, Jorge Finke, and Camilo Rocha. "Feature extraction with spectral clustering for gene function prediction using hierarchical multi-label classification". In: *Applied Network Science* 7.1 (Dec. 2022). Number: 1 Publisher: SpringerOpen, pp. 1–20. ISSN: 2364-8228. DOI: 10.1007/s41109-022-00468-w.

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation.* 2015. arXiv: 1505.04597 [cs.CV].

[26] Jose J. Salazar, Lean Garland, Jesus Ochoa, and Michael J. Pyrcz. "Fair train-test split in machine learning: Mitigating spatial autocorrelation for improved prediction accuracy". In: *Journal of Petroleum Science and Engineering* 209 (Feb. 1, 2022), p. 109885. ISSN: 0920-4105. DOI: 10.1016/j.petrol.2021.109885.

[27] Tomaž Stepišnik and Dragi Kocev. "Hyperbolic Embeddings for Hierarchical Multi-label Classification". In: *Foundations of Intelligent Systems.* Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 66–76. DOI: 10.1007/978-3-030-59491-6_7.

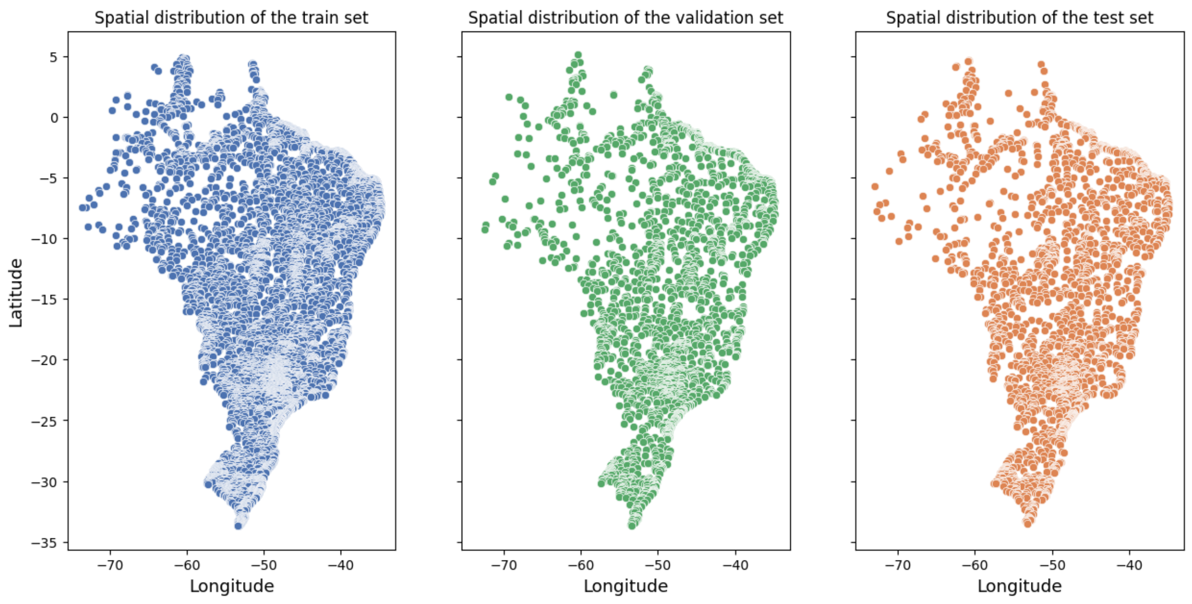
[28] Mingxing Tan and Quoc V. Le. "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks". In: (2020). arXiv: 1905.11946 [cs.LG].

[29] *The European space agency.*

[30] Mehmet Ozgur Turkoglu, Stefano D'Aronco, Gregor Perich, Frank Liebisch, Constantin Streit, Konrad Schindler, and Jan Dirk Wegner. "Crop mapping from image time series: Deep learning with multi-scale label hierarchies". In: *Remote Sensing of Environment* 264 (2021), p. 112603. ISSN: 0034-4257. DOI: https://doi.org/10.1016/j.rse.2021.112603.

Appendix A

Supplementary materials



Spatial distribution of the train set
Spatial distribution of the validation set
Spatial distribution of the test set
Latitude
Longitude
5
0
−5
−10
−15
−20
−25
−30
−35
−70
−60
−50
−40

Figure A.1: Spatial distribution of the training, validation and test sets

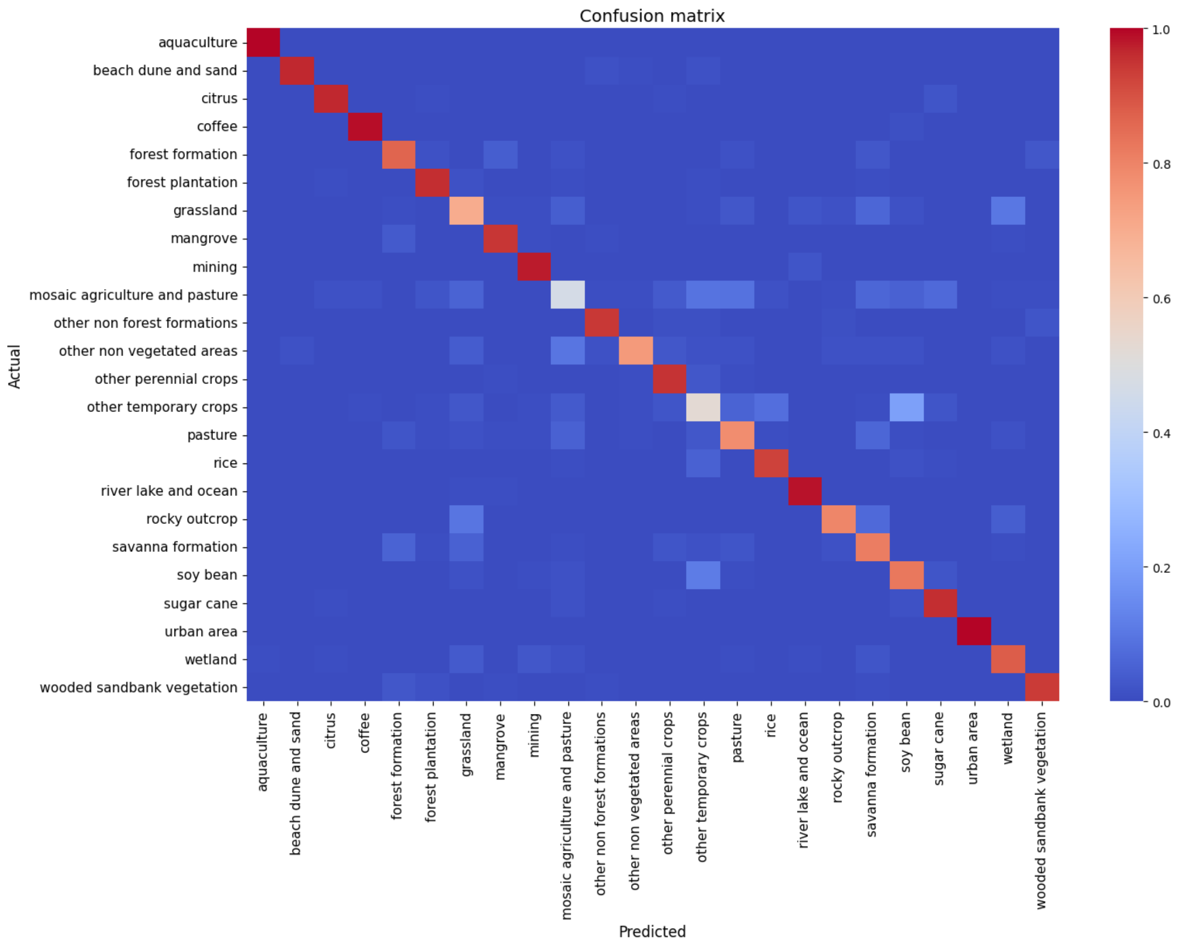


Figure A.2: Confusion matrix on leaf labels from *custom-Arch 2* trained with MCLoss. The most difficult classes to classify are "mosaic agriculture and pasture" and "other temporary crops".

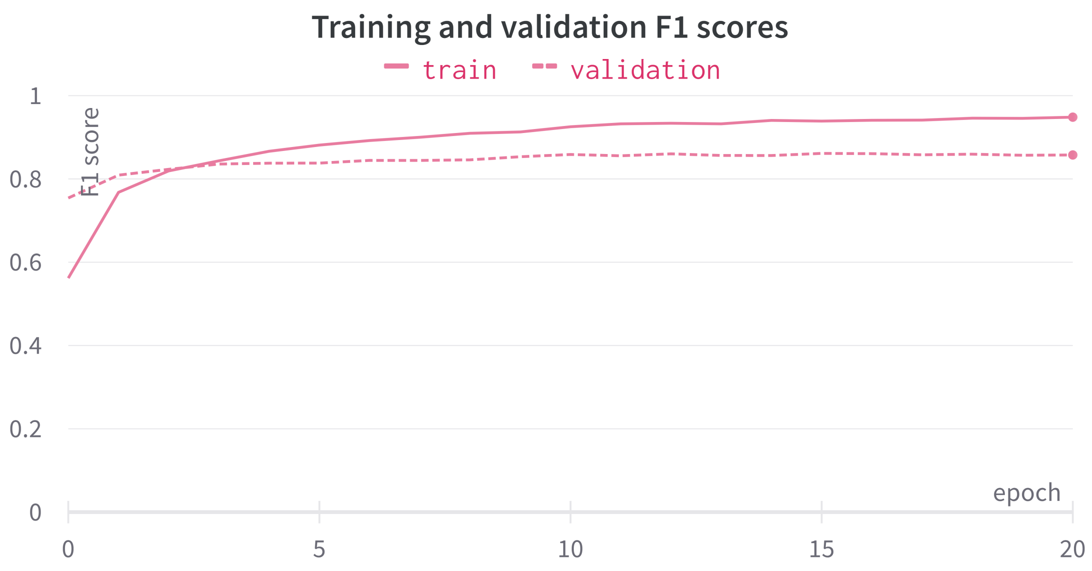
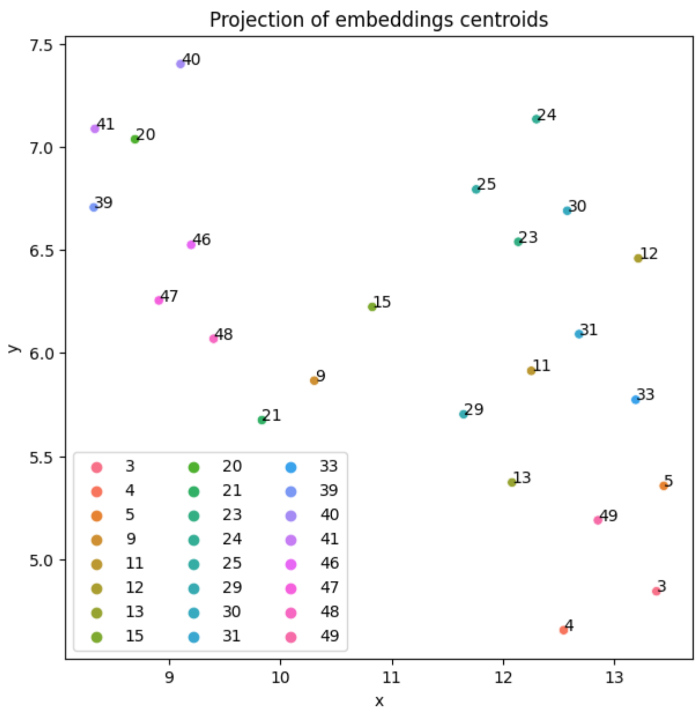
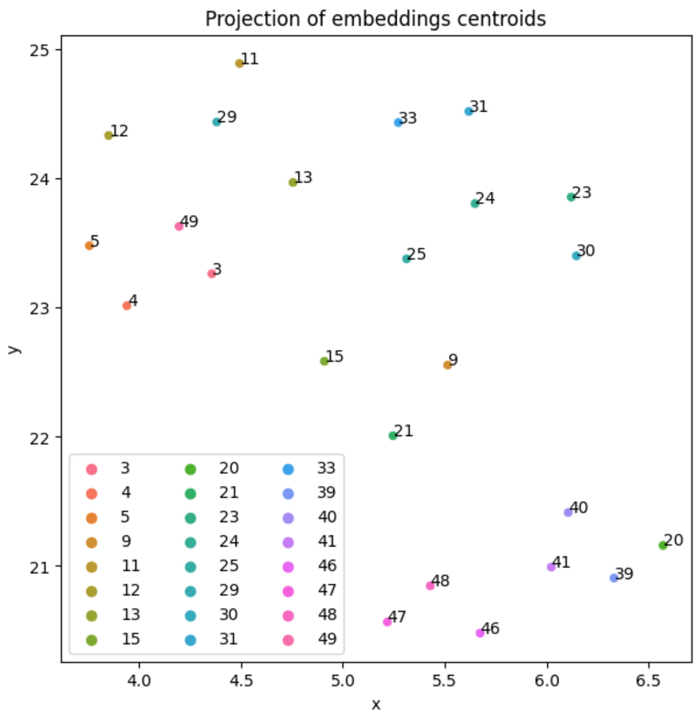


Figure A.3: Macro averaged F_1 scores on training and validation sets for the baseline model. Hierarchical multi-label classification models had a similar trend.



(a) Projected embeddings learnt with the most performing model on leaf labels (see results in bold in table 5.1).



(b) Projected embeddings learnt via semi-supervised learning (using Eq. 4.17).

Figure A.4: Visualization of class centroids embeddings on the test set. The embeddings were projected using UMAP [17]. The labels are presented instead of the actual class names which are visible in figure 3.1.