POLITECNICO DI TORINO

Master's Degree in Computer Engineering

Master's Degree Thesis

Aboveground Biomass Estimation from Satellite Imagery

Supervisors

Prof. Paolo GARZA

Dr. Edoardo ARNAUDO

Dr. Luca BARCO

Candidate

Francesco SCALERA

December 2023

Abstract

In the field of studying carbon release and sequestration by forests, Aboveground Biomass (AGBM) is a popular measure for estimating how much carbon dioxide forests are absorbing and releasing into the atmosphere. Remote sensing methods provide a faster and less destructive approach for biomass estimation compared to the more invasive technique called destructive sampling, specifically the permanent removal of material from a specimen for analyses. Additionally, this information enables landowners and policymakers to make more informed decisions for forest conservation.

In this context, LiDAR is a remote sensing technique that offers precise information about the Earth's surface. When combined with on-the-ground sampling methods, it allows for accurate biomass estimation. Unfortunately, the main disadvantages lie in the cost and time consumption associated with collecting data using airborne LiDAR. On the other hand, Sentinel-1 and Sentinel-2 satellite imagery provide a variety of spectral bands that can be effectively used to measure and derive different metrics for forest management.

The objective of this work is to leverage deep learning techniques to estimate the annual biomass of various sections in Finland's forests using imagery from Sentinel-1 and Sentinel-2, with ground truth data derived from airborne LiDAR surveys and in-situ measurements. To achieve this, two models are proposed and implemented: one is Swin UNETR, a Transformer-based UNet, and the other is an Attention Unet: a Unet with a shared encoder that uses aggregation via attention. Various experiments are conducted involving different types of loss functions for training. Both models achieve good results in terms of Root Mean Square Error (RMSE), considered as evaluation metric, demonstrating their good performance in the biomass estimation task.

Acknowledgments

Vorrei ringraziare il prof. Paolo Garza e il dott. Edoardo Arnaudo di LINKS Foundation per la loro disponibilità e per avermi dato l'opportunità di svolgere al meglio questo lavoro. I miei più grandi ringraziamenti anche al dott. Luca Barco che con la sua gentilezza, pazienza e disponibilità è stato pronto a risolvere ogni mio dubbio ed è riuscito a guidarmi sapientemente a raggiungere gli obiettivi della tesi.

Grazie a mamma e papà per il loro amore incondizionato e il loro supporto continuo. Forse potrei solamente ringraziarvi per aver lavorato tanto su di me, sacrificando tutto, affinché questo momento arrivasse. Ma oltre a questo, volevo dirvi che ciò che di più grande mi avete donato è la libertà. La libertà di scoprire chi sono, di seguire il mio percorso, di commettere errori, di coltivare passioni e raggiungere obiettivi, che spesso sono diventati anche i vostri. Un giorno spero di trasmettere a qualcuno anche solo una piccola parte di ciò che voi avete trasmesso a me.

Grazie a Simone. L'essere diversi, spesso in contrasto, non scalfisce il legame indissolubile che c'è tra noi due, ma anzi lo rafforza. Mi hai sempre offerto spunti di discussione e confronto ed hai sempre cercato di tendermi la mano nei momenti in cui mostravo le mie fragilità, che probabilmente sei uno dei pochi a conoscere profondamente.

Grazie ad Andrea, Gian Paolo, Giuseppe, Carmine e Peppe. Appena arrivato a Torino mi sono chiesto se qui sarei riuscito a coltivare dei rapporto profondi con qualcuno, lontano dal luogo dove sono cresciuto e dagli ambienti che ho abitualmente frequentato. In voi ho trovato una famiglia, con cui ho condiviso tutto. Tutte le esperienze vissute in questi anni, dalle più belle alle più brutte, rimarranno per sempre ricordi vivi nella mia mente solo perché, in fondo, con me c'eravate voi. Vi voglio bene.

Grazie a Matteo. Non siamo mai stati due persone che hanno comunicato l'affetto reciproco tramite tante parole, abbiamo intrapreso due strade molto diverse e per questo capitano spesso periodi in cui non ci non sentiamo. Nonostante ciò, la nostra amicizia è sempre stata una costante della mia vita e non ho mai dubitato del fatto che mi sarai sempre vicino, proprio come fece come quel bambino col cappellino

del Milan il primo giorno di scuola media.

Grazie a tutti gli amici di Mesagne. Il fatto che voi siate qui per condividere questo momento con me mi rende orgoglioso di avervi conosciuto e ancora più consapevole dell'importanza che ricopre il nostro rapporto.

Grazie a tutte le altre persone incontrate durante il percorso, anche quelle che ne hanno condiviso con me solo una parte: da tutti avrò sicuramente tratto un insegnamento per arricchirmi. Spero di essere importante per voi almeno quanto lo siete per me.

Non ci saranno ringraziamenti a me stesso. Questo traguardo rappresenta per me la sintesi di tutto quello che le persone mi hanno trasmesso e le esperienze che con loro ho vissuto. Siete stati tutti per me fonte di ispirazione ed è solo grazie a voi se questo momento rimarrà un ricordo indelebile.

Table of Contents

List of Tables VIII

List of Figures IX

1 Introduction 1

2 Related Works and Backgrounds 4
2.1 UNet . . . 4
2.2 Attention UNet . . . 6
2.3 Swin Transformer . . . 6
2.4 Swin UNETR . . . 8
2.5 Focal Frequency Loss . . . 9

3 Dataset 12
3.1 Data Source . . . 12
3.1.1 Sentinel-1 . . . 12
3.1.2 Sentinel-2 . . . 14
3.1.3 LiDAR ground truth data . . . 16
3.2 Data processing . . . 16
3.2.1 Dataset information . . . 17
3.2.2 Data preprocessing . . . 18

4 Methodology 21
4.1 Problem statement . . . 21
4.2 Network architectures . . . 21
4.2.1 UNet with Attention Pooling . . . 21
4.2.2 Swin UNETR . . . 22
4.3 Loss functions . . . 24
4.3.1 Root Mean Squared Error . . . 24
4.3.2 Mean Absolute Error . . . 25
4.3.3 Structural Similarity Index . . . 25

4.3.4 Focal Frequency Loss 26

5 Experiments 28
5.1 Implementation details 28
5.1.1 Data preparation 28
5.1.2 Scenarios definition 29
5.1.3 Multi-GPUs training setup 30
5.1.4 Training process 30
5.2 Evaluation metric 34
5.3 Results 34

6 Conclusions 39

Bibliography 40

List of Tables

3.1 Description of Sentinel-1 Bands.[26] 13
3.2 Description of Sentinel-2 Bands. 14
3.3 Description of fields in metadata file.[27] 17
3.4 Summary of data statistics for Sentinel-1 and Sentinel-2 data sources. 19
3.5 Summary of data statistics for ground-truth data. 20

4.1 Description of Swin UNETR's hyperparameters. 24

5.1 Sample distribution for training, validation, and test sets. 29
5.2 Summary of experiments' scenarios. 30
5.3 Package list with associated version. 32
5.4 Summary of hyperparameters' values. 33
5.5 Performance metrics for different experiment scenarios and folds in terms of RMSE. 35

List of Figures

2.1 UNet architecture.[14] 5
2.2 Attention UNet architecture.[11] 6
2.3 Overall architecture of Swin Transformer model.[13] 7
2.4 Window partition mechanism between two consecutive layer.[13] . . 7
2.5 Overall architecture of Swin UNETR, designed for segmentation of 3D multi-modal MRI images.[12] 9
2.6 Frequency distanze between two vectors $\vec{r_p}$ and $\vec{r_{gt}}$.[22] 11

3.1 Sentinel-1 images for chip 001b0634, by band.[27] 13
3.2 Sentinel-2 images for chip 001b0634, by band.[27] 15
3.3 LiDAR image for chip 001b0634.[27] 16

4.1 Overview of the Swin UNETR model implemented. 23

5.1 Example of learning rate schedule employed in Scenario 2 experiments. 31
5.2 Comparison between ground-truth data and Swin UNETR's outputs. 37
5.3 Comparison between frequency spectra of ground truth and Swin UNETR's outputs. 38

Chapter 1

Introduction

Depending on their characteristics and local circumstances, forests can play different roles in the carbon cycle, from net emitters to net sinks of carbon. Forests sequester carbon by capturing carbon dioxide from the atmosphere and transforming it into biomass through photosynthesis. Sequestered carbon is then accumulated in the form of biomass, deadwood, litter and in forest soils. Release of carbon from forest ecosystems results from natural processes (respiration and oxidation) as well as deliberate or unintended results of human activities (i.e. harvesting, fires, deforestation).[1]

Where carbon sequestration prevails, forests helps to reduce carbon emissions by acting as a mechanism to sequester additional carbon with the creation of a sink that it has previously retained. On the contrary, if the net of carbon emission by forests is positive, forest accentuate and accelerate greenhouse effect processes, contributing significantly to climate change.

On the other hand, climate change also impacts forest health. Rising temperatures and rainfall, which could also bring beneficial effects to forests, cause uncertainty about the role of forests in the carbon cycle as it could accelerate some processes or could mitigate some negative factors.

To measure how much forests behavior change over time, scientists looks to estimate Aboveground Biomass (AGBM). This is a widespread measure in the study of carbon release and sequestration by forests, in order to understand how much carbon a forest contains (its carbon stock) and how it changes (carbon flux). This information plays an important role in making better decisions for the conservation of forest by landowners and policy makers.[2] In more details, Aboveground Biomass includes all biomass in living vegetation, both woody and herbaceous, above the soil including stems, stumps, branches, bark, seeds and foliage. Variations in it represents a key sign that the environment changes for the effect of policies or benefits associated to carbon emissions.[3]

Over the years, various methods have been used to estimate this index, such as

destructive sampling methods, which are damning and very invasive techniques for the environment: they consist of cutting a sample of trees and vegatation to measure certain characteristics.[4] Although they provide for very accurate measurements, coupled with the fact that it is a very pratical and cost-effective method, these procedures cause extensive and permanent damage to the specimens that are involved in the measurements.

Thus, efforts are being made to use fewer of the previously described techniques in favor of remote sensing methods. In this sense, these methods offer a faster and less destructive method of estimating AGBM in an extensively more significant geographic area.[2] For example, measurements using LiDAR methods generate very accurate three-dimensional measurements about the Earth's surface.[2] A LiDAR device, that is composed principally of a laser, a scanner, and a specialized GPS receiver, uses the light in form of pulsar for its measures.[5]

Other important data for measuring vegetation and environmental health come from the Sentinel-1 [6] and Sentinel-2 [7] space missions. They provide satellite imagery by collecting information via C-band SAR [8] and MSI [9], respectively. In this regard, by capturing bands at different frequencies, various phenomena such as land use changes, winds and waves can be monitored.[2] In addition, satellites imagery are more timely and has wider coverage with respect to LiDAR.

Awareness regarding the importance of the state of forests and the fight against climate change has significantly helped research focus on such issues. For example, in 2021 DrivenData [10] held a competition with the goal of providing a deep learning and machine learning algorithm that estimates Aboveground Biomass annually of Finnish forests. The data provided for estimation are satellite imagery from Sentinel-1 and Sentinel-2 satellites that are used as input data for the algorithm, while ground truth comes from LiDAR sensors combined with in-situ measurements.[2]

The goal of this thesis is therefore to try to contribute to the cause to offer a method for predicting the conservation status of forests. For this purpose, deep learning techniques were employed to process the available data, which came from the competition mentioned earlier. The work then took the form of solving a semantic segmentation task in which two very effective models were employed to provide a yearly AGBM prediction over each patch of forest to process.

In fact, after carefully analyzing and processing the images in a way that made it easier to train the deep learning models and at the same time trying to retain as much information as possible, two architectures were identified that demonstrate effectiveness in dealing with the type of task configured. One is a modified version of the Attention UNet [11] model; the use of Attention modules was particularly useful in aggregating the temporal dimension of the input. On the other hand, it was chosen to implement Swin UNETR [12], in which the so-called "U-shaped" architecture is made explicit using a Transformer-based backbone such as Swin Transformer [13]. The experiments conducted involved different training strategies

for the models, trying to minimize several loss functions that could prove effective in achieving acceptable results.

After this brief introduction, the structure of the work consists of Chapter 2, which includes an in-depth analysis on the background and related works specific to the semantic segmentation tasks, which helped in the implementation choices, while Chapter 3 focuses more on the description of the image data at hand, describing their origin and the processing techniques performed. Chapter 4 presents in detail the architectures and loss functions employed in the work. The setups and different scenarios involved in different experiments conducted are described extensively in Chapter 5, where the results achieved are also reported and discussed there. Finally, in Chapter 6, a general discussion of all the work is provided with interesting hints on future work that could be done to improve the results obtained.

Chapter 2

Related Works and Backgrounds

This chapter presents an introduction to the main deep learning techinques adopted for this work. Firstly, it introduces a general presentation of the two models adopted for the task, Swin UNETR [12] and Attention UNet [11], and consequently it continues with a presentation of the Focal Frequency Loss Function implemented for a handful experiments.

2.1 UNet

The UNet architecture [14] is a convolutional neural network (CNN) that is frequently used for semantic segmentation tasks in biomedical image analysis, satellite imagery, and other domains where exact delineation and understanding of object boundaries inside pictures is critical. The name "UNet" is derived from the network's U-shaped architecture that consists of an expansive path and a contractive path, as highlited in Figure 2.1, where the typical architecture is shown, mimicking an encoder-decoder structure.

The expansive path has the major goal to extract hierarchical and abstract features from the input image while gradually lowering its spatial dimensions.[14] The encoder progressively transforms the input into a higher-level feature space, where each subsequent layer captures more abstract and complex features compared to the previous layers. This hierarchical representation is fundamental for successful semantic segmentation. It typically consists of multiple convolutional and pooling layers that progressively reduce the spatial dimensions and comprise informations of the input while extracting features and helping the network to discard.[15]

On the other hand, the "decoder" path is the counterpart to the contractive path

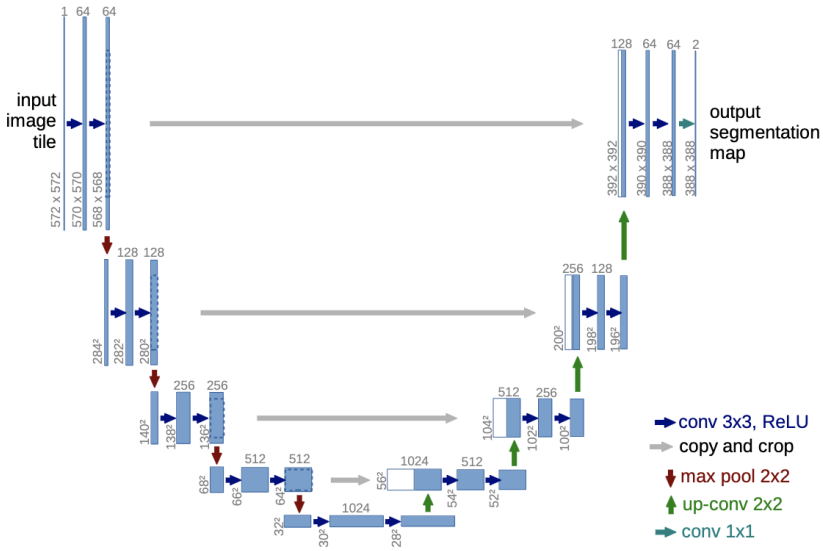

Figure 2.1: UNet architecture.[14]

(encoder). It plays a critical role in the UNet's ability to generate the final high-resolution segmentation map from the abstract and reduced-dimensional feature representation obtained by the encoder. As it expands the feature maps, it aims to recover and reconstruct spatial information that was lost during the contracting path. This is achieved by gradually increasing the spatial resolution to match the original input size. The decoder path reconstructs the high-resolution segmentation map from the features obtained by the encoder. It comprises upsampling layers, usually transposed convolutions or upsampling followed by convolutional layers, to gradually increase the spatial dimensions back to the original size.

The process of expanding the feature maps back to the original spatial dimensions of the input image typically involves reversing the reduction performed by the encoder through up-sampling operations, such as transposed convolutions or simple interpolation techniques.[15] Another important key features of the Unet structure are the skip connections, which link the corresponding levels from the encoder to the decoder path. The network may integrate low-level and high-level features for improved segmentation thanks to these connections, which let the decoder incorporate high-resolution information from the contracting path with the upsampled feature maps. The decoder path enables accurate localization by utilizing data from the skip connections. In order to produce the final segmentation map with greater accuracy, it refines the segmented regions and makes use of the hierarchical features.

2.2 Attention UNet

Convolutional Neural Networks (CNNs) have become known as an important tool in image processing for a variety of applications. However, they show certain limitations, such as the incapacity to capture long-range contextual information and sensitivity to image noise. To overcome these issues, a concept called "attention mechanism" has become known, firtsly introduced as an improvement over the encoder-decoder based systems in Natural Language Processing (NLP), then adapted in other various domains, like computer vision.

In this field, attention mechanisms are used to allow the model to focus on relevant regions of the input image, in order to better capture local and global data relations and improve the model's capacity for adaptive information processing. Furthermore, the introduction of the mechanisms contribute to better network generalization and lost of fewer computational resources on irrelevant activations.[16]

One of the architecture that exploits these concepts is Attention UNet [11]. As shown in Figure 2.2, firstly it incorporates a UNet architecture [14], which consists of an encoding path for feature extraction and a decoding path for generating pixel-wise predictions. Secondly, it integrates attention modules within the architecture, which dynamically modulate the importance of different spatial locations in the feature maps.

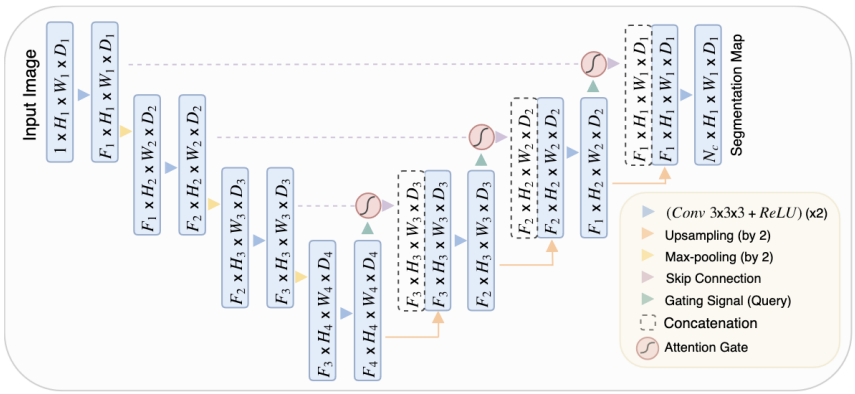

Figure 2.2: Attention UNet architecture.[11]

2.3 Swin Transformer

In recent periods, the Transformer [17] model has become firmly established in the field of natural language processing (NLP) because it is very effective attention mechanism to model long-range dependencies in the data. Research over the years

has then developed by trying to adapt the concepts introduced by them in the area of Computer Vision, where excellent results have been demonstrated in certain types of tasks such as image classification and joint vision-language modeling.[18]

One of the major problems to be solved in order to apply Transformers in computer vision is that tokens, which are the basic element of Transformers in NLP, can vary greatly in scale, which is not the case in language models. Another issue to overcome lies in lowering the complexity of self-attention computation for images, which is quadratic with respect to image size, especially in semantic segmentation tasks where prediction is performed at the pixel level.[13] For these reasons, the Swin Transformer [13] architecture was introduced, which succeeds in generating hierarchical feature maps and has linear computational complexity to image size. Specifically, the hierarchical representation is built from small patches, which are gradually merged into the deeper layers of the architecture. Because of this hierarchical design, which allows the model to capture local and global context, it is very easy to consider this architecture as a backbone for models that operate dense prediction such as UNet. An overview of the presented architecture is available in Figure 2.3.

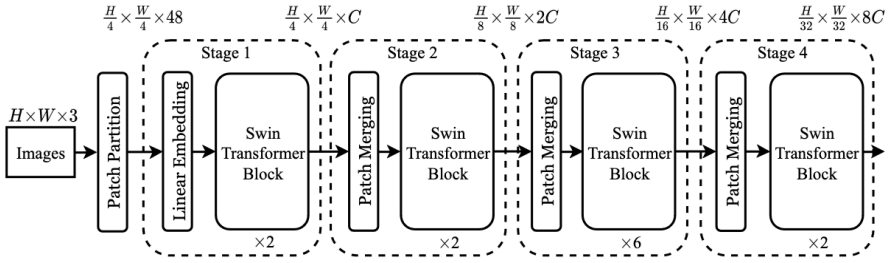

Figure 2.3: Overall architecture of Swin Transformer model.[13]

Linear complexity is achieved through the implementation of non-overlapping windows that partition an image, as is highlighted in Figure 2.4.

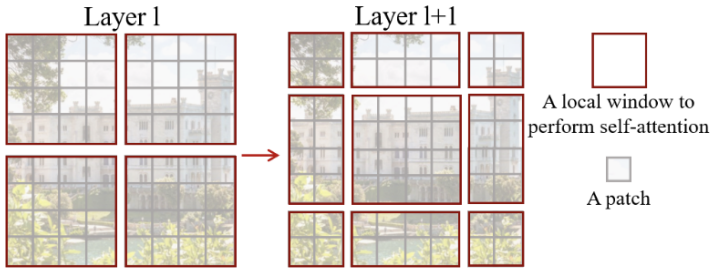

Figure 2.4: Window partition mechanism between two consecutive layer.[13]

Because the number of patches in each window is fixed, the complexity is proportional to image size. Another important aspect relies on the shifting mechanism of window partition between two consecutive self-attention layers: the shifted windows connect the windows of the previous layer, giving connections that considerably increase modeling capability and facilitates memory access in hardware. In contrast to prior Transformer-based designs like [19], which output feature maps of a single resolution and have quadratic complexity, Swin Transformer is suitable as a general-purpose backbone for many vision applications.

2.4 Swin UNETR

Transformer models have shown impressive abilities to capture extensive information across various domains, such as natural language processing and computer vision. In the context of computer vision, Vision Transformers (ViTs) have achieved remarkable performance on various benchmarks. This success is attributed to their self-attention module, which facilitates the modeling of long-range information through pairwise interactions among token embeddings. Consequently, this approach results in more efficient representation of both local and global context. [19, 18]

In this regard, UNETR architecture [20], that draws inspiration from a populaur "U-shaped" UNet architecture [14], it is the first model that implement ViT as encoder, in order to better capture spatial relationships in images. More recently, Swin Transformer [13] have been proposed as a hierarchical vision transformer that computes self-attention in an efficient shifted window partitioning scheme, enabling efficient processing of images of varying sizes.

Swin UNETR [12] represents a model that combines the two aforementioned approaches: in fact, it consists in a U-shaped Network with a Swin Transformer [13] as encoder, connected to a CNN-based decoder at different resolutions via skip connections. With the help of the Figure 2.5, that describe explicitly the original implementation of SwinUNETR in [12] designed for 3D multi-modal MRI images, it can be seen that long-range data relationships can be captured by Swin Transformer thanks to its multi-head attention capabilities, and its UNETR structure guarantees that spatial information is effectively retained during processing. This architecture has demonstrated outstanding performance in a variety of semantic segmentation tasks in computer vision.

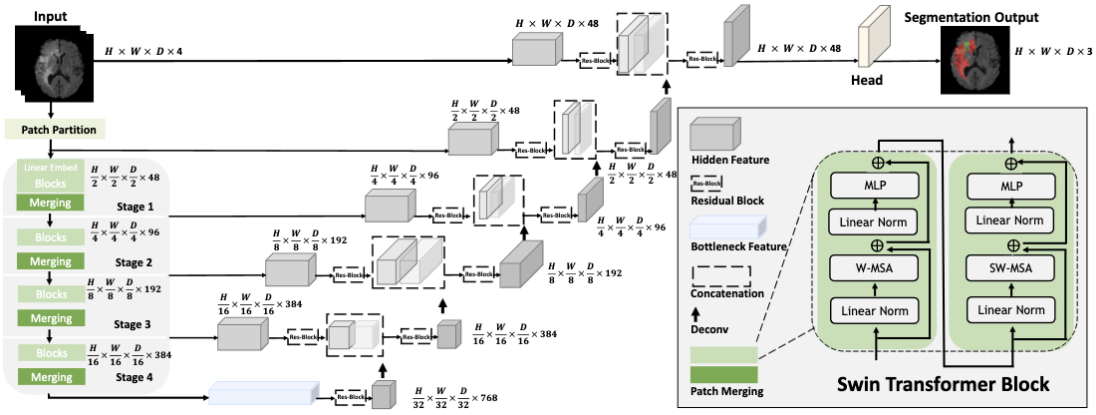

Figure 2.5: Overall architecture of Swin UNETR, designed for segmentation of 3D multi-modal MRI images.[12]

2.5 Focal Frequency Loss

The remarkable progress and the immense success in recent years of generative models highlights an open issue concerning the gaps between real and generated images. In some cases, these gaps may only be revealed through the frequency spectrum analysis: in reconstruction and synthesis tasks may be imputed to a learning bias of neural network, called *spectral bias* [21], that is evidenced by Fourier analysis. In this regard, generative models tend to eschew frequency components that are hard to synthesize, i.e., hard frequencies, and converge to an inferior point.

Focal Frequency Loss [22] allows a model to adaptively focus on frequency components that are hard to synthesize by down-weighting the easy ones. This objective function is complementary to existing spatial losses, offering great impedance against the loss of important frequency information due to the inherent bias of neural networks.

Specifically to this work, since that is not a task of reconstruction and synthesis, we want analyze and close the gap between model's output image and the ground truth in the frequency domain.

Towards this goal, the usual discrete Fourier transform (DFT) is applied to both samples to obtain their frequency representations.

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{i2\pi(\frac{ux}{M} + \frac{vy}{N})} \tag{2.1}$$

Where:

- The image size is $M \times N$

- (x, y) denotes the coordinate of an image pixel in the spatial domain
- $f(x, y)$ is the pixel value
- (u, v) represents the coordinate of a spatial frequency on the frequency spectrum
- $F(u, v)$ is the complex frequency value;

Following Euler's formula:

$$e^{i\theta} = \cos\theta + i\sin\theta \tag{2.2}$$

The exponential funtion in 2.1 can be rewritten as:

$$e^{i2\pi(\frac{ux}{M}+\frac{vy}{N})} = \cos 2\pi(\frac{ux}{M} + \frac{vy}{N}) + i\sin 2\pi(\frac{ux}{M} + \frac{vy}{N}) \tag{2.3}$$

Due to to periodicity of trigonometric functions (highlighted in 2.3), images show periodic qualities when broken down into sines and cosines and every coordinate value on the frequency spectrum, which represents a particular spatial frequency, is dependent upon every image pixel in the spatial domain.[22]

Each spectrum coordinate (u, v) value is mapped to a Euclidean vector in a two-dimensional space, with both the amplitude and phase information of the spatial frequency put under consideration. Assuming that $F_p(u, v)$ be the spatial frequency value at the spectrum coordinate (u, v) of the models' prediction, and the corresponding $F_{gt}(u, v)$ has the same meaning for the ground-truth, we map from these values two vectors: $\vec{r_{gt}}$ and $\vec{r_t}$. The proposed loss function is defined by the scaled Euclidean distance of these vectors:

$$d(\vec{r_{gt}}, \vec{r_p}) = \|\vec{r_{gt}} - \vec{r_p}\|_2^2 = |F_{gt}(u, v) - F_p(u, v)|^2 \tag{2.4}$$

At this point, the frequency distance between the two images can be written as the average value over each pixel:

$$d(F_p, F_{gt}) = \frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}|F_{gt}(u, v) - F_p(u, v)|^2 \tag{2.5}$$

In order to down-weight easy frequencies, it is added to the formulation a dynamic spectrum weight matrix, with each element denoted as $w(u, v)$:

$$w(u, v) = |F_{gt}(u, v) - F_p(u, v)|^\alpha \tag{2.6}$$

Where α represents a scaling factor. Intuitively, the matrix is updated on the fly according to a non-uniform distribution on the current loss of each frequency during training. Explicitly minimizing the distance of coordinate values on the

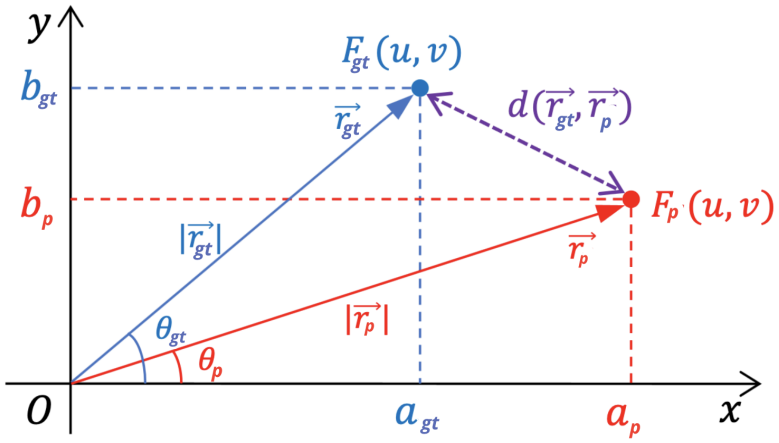

Figure 2.6: Frequency distanze between two vectors $\vec{r_p}$ and $\vec{r_{gt}}$.[22]

samples' spectra can assist networks in quickly locating and focus tough portions of the spectrum, i.e., hard frequencies. At the end, the final formulation of the Focal Frequency Loss (FFL) is:

$$d(F_p, F_{gt}) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} w(u,v) |F_p(u,v) - F_{gt}(u,v)|^2 \tag{2.7}$$

Chapter 3

Dataset

3.1 Data Source

In recent years there has been a need to monitor the state of the Earth and the environment, to mitigate the problem of climate change and monitor the possibility of natural disasters. For this reason, European Commission (EC), jointly with European Space Agency (ESA) [23], established the Copernicus programme [24] for the implementation of information services dealing with environment and security.

These information are based on satellite and in-situ measurements for delivering near-real-time open access data on a global level which can be used for local and regional need. In reference to the firsts, Copernicus is served by a set of satellites, the Sentinel family, engineered to operate in a pre-programmed, conflict-free operation mode, providing high-resolution imagery of every content on Earth, as well as coastal zones, shipping routes, and the entire ocean.[6]

Specifically to this work, image data that comes from satellites Sentinel-1 and Sentinel-2 are chosen as input for implemented models. This work also deals with LiDAR airborne data (combined with in-situ methods) to collect ground-truth precise Aboveground Biomass (AGBM) measurements.

The dataset of images that was used in this work came from "The BioMassters" [2] competition held by DrivenData [10].

3.1.1 Sentinel-1

The Copernicus Sentinel-1 (S1) mission of the European Space Agency consists of a constellation of polar-orbiting satellites that operate day and night to carry out C-band Synthetic Aperture Radar (SAR) imagery.[6] Thanks to this technique, satellites are allowed to collect data on a site in any weather condition because they operate at wavelengths that are unaffected by clouds or lack of illumination. SAR satellites migrate from the north pole towards the south pole for half of their

trajectory. This direction is called descending orbit. On the other hand, when the satellite is travelling from south pole to north pole, it is referred to be in ascending orbit.[2]

Since the mission is composed by two satellites, Sentinel-1A and Sentinel 1-B, data comprises two bands ("VV" and "VH") for each satellites, for a total of four bands. A description of Sentinel-1 bands is provided in in Table 3.1, while in Figure 3.1 are presented a visual representation of S1 bands employed in this work. These bands are captured from sensors that emits energy via vertically polarized signals (represented by the first "V" in bands' name), and then they record the amount of energy reflected back after interaction with Earth, receiving either vertically (V) or horizontally (H) polarized signals. Thanks to this aspect, scientists can learn more about the surface of the region being sensed by analyzing the signal strength of these various polarizations.[25]

Band	**Description**	**Resolution**
VV ascending	C-band SAR	10m
VH ascending	C-band SAR	10m
VV descending	C-band SAR	10m
VH descending	C-band SAR	10m

Table 3.1: Description of Sentinel-1 Bands.[26]

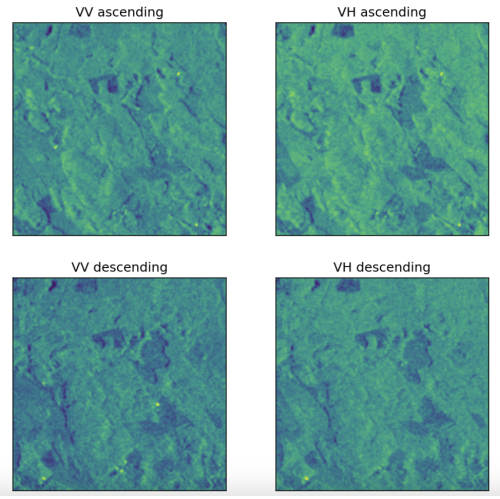

Figure 3.1: Sentinel-1 images for chip 001b0634, by band.[27]

3.1.2 Sentinel-2

Sentinel-2 (S2) mission provides wide-swath, high-resolution, multi-spectral imagery to monitor vegetation, soil, water cover, inland waterways and coastal areas. The mission includes two satellites that orbit with the same trajectory but phased at 180°: this allows for a high revisit frequency of five days at the Equator.

S2 satellites measurements are provided by a Multispectral Instrument (MSI) that collects data in the visible, near-infrared, and short-wave infrared portions of the electromagnetic spectrum. This instrument contributes to land studies and provides valuable data for land cover classification, atmospheric correction and cloud masks.[7]

In particular, the optical instrument samples 13 spectral bands, and the principal difference compared to Sentinel-1 is that SAR instrument uses wavelengths that range from centimeters to meters, while Sentinel-2's MSI measures shorter spectral bands that range from 400 to 2400 nanometers. In Table 3.2 is reported a summary of bands collected, with the correspondent resolution and a brief description. It's important to pay attention on CLP band, that represents cloud probability: since S2 sensors cannot penetrate clouds, this layer indicates the percentage probability of cloud cover for each pixel, and it has values that range from 0 to 100.[27]

In addition, Sentinel-2 bands were used because they provide higher resolution than those captured by Sentinel-1.

Band	Description	Resolution
B1	Coastal aerosol, 442.7 nm (S2A), 442.3 nm (S2B)	60m
B2	Blue, 492.4 nm (S2A), 492.1 nm (S2B)	10m
B3	Green, 559.8 nm (S2A), 559.0 nm (S2B)	10m
B4	Red, 664.6 nm (S2A), 665.0 nm (S2B)	10m
B5	Vegetation red edge, 704.1 nm (S2A), 703.8 nm (S2B)	20m
B6	Vegetation red edge, 740.5 nm (S2A), 739.1 nm (S2B)	20m
B7	Vegetation red edge, 782.8 nm (S2A), 779.7 nm (S2B)	20m
B8	NIR, 832.8 nm (S2A), 833.0 nm (S2B)	10m
B8A	Narrow NIR, 864.7 nm (S2A), 864.0 nm (S2B)	20m
B9	Water vapour, 945.1 nm (S2A), 943.2 nm (S2B)	60m
B11	SWIR, 1613.7 nm (S2A), 1610.4 nm (S2B)	20m
B12	SWIR, 2202.4 nm (S2A), 2185.7 nm (S2B)	20m
CLP	Cloud probability, based on s2cloudless	160m

Table 3.2: Description of Sentinel-2 Bands.

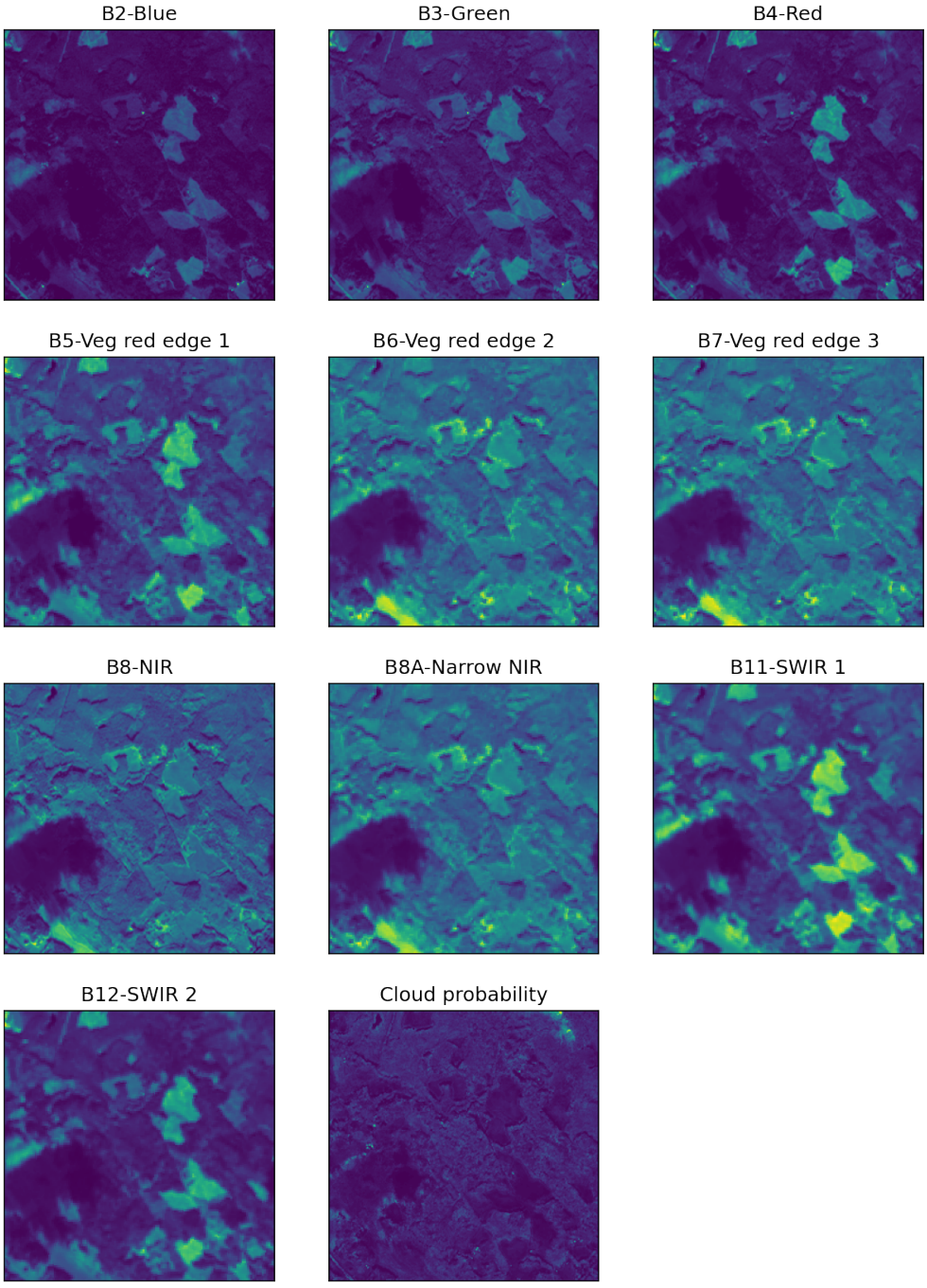

Figure 3.2: Sentinel-2 images for chip 001b0634, by band.[27]

3.1.3 LiDAR ground truth data

LiDAR, acronym for Light Detection and Ranging, is a remote sensing technique that provides 3D information about the terrain and vegetation by measuring variable distances to the Earth with light in form of pulsed laser. In particular, when the laser points a targeted area on the ground, the beam of light is reflected by the surface it encounters. The reflection is collected with a sensor that, in combination with position and orientation data generated from integrated GPS and Inertial Measurement Unit systems, generate detail-rich point clouds. These point clouds are then used to generate precise and three-dimensional informations about Earth surface and shape.[5]

The main components of a LiDAR instrument are a laser, a scanner and a GPS receiver, with airplanes and helicopters that are the most widely used platforms to collect LiDAR data over a broad area. Scientists and cartographers may analyze natural and artificial settings with flexibility, accuracy, and precision thanks to LiDAR equipment.

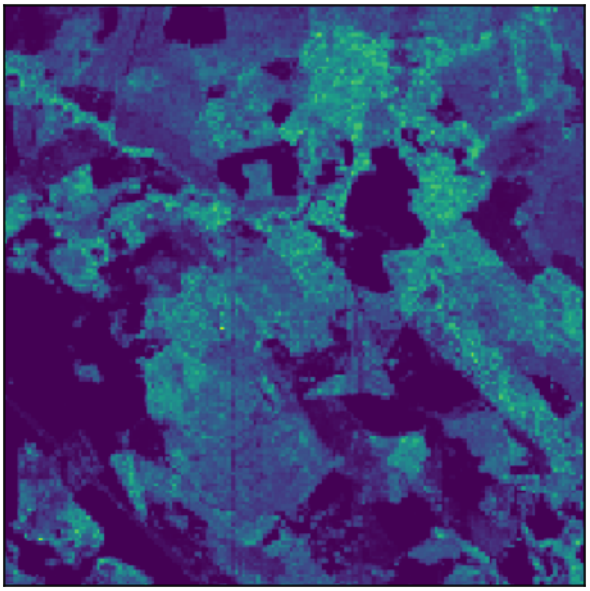

Figure 3.3: LiDAR image for chip 001b0634.[27]

3.2 Data processing

The data for this thesis is imagery collected by Sentinel-1 and Sentinel-2 satellites, together with LiDAR airborne AGBM measurements for Finnish forests. In the

following subsections, a detailed explanation of how the feature data are gathered to input the deep learning models implemented in this work (more details in Chapter 4), followed by a presentation of pre-processing methods applied to data.

3.2.1 Dataset information

The input dataset is composed from images coming from Sentinel-1 and Sentinel-2 missions for 8,689 patches of forest in Finland. Each patch, or chip, represents an area of forest of 2,560 × 2,560 square meters. To be compliant with Sentinel-2 bands resolution, these images have been resized to 10 meters resolution by using geometric and radiometric corrections, for this reason each image is 256x256 pixels in size, with each pixel that represents an area portion of 10 square meters. Images are monthly aggregations delivered as GeoTIFFs with any associated geolocation data deleted.[27] It is highlighted that of the 13 bands that Sentinel-2 samples, only 11 were used in subsequent experiments: these bands are B2, B3, B4, B5, B6, B7, B8, B8A, B11, B12, and CLP.

For every patch in the dataset, there are captures that cover an entire year period, specifically from September to August. It is now evident that the dataset should includes 24 images associated with the same patch, 12 of which originate from Sentinel-1 and the remaining 12 from Sentinel-2, but there are some of them that do not have full year coverage due to some outage, which is why some data for some months are missing.

Information about each satellite image, including its corresponding patch, satellite, and the month in which it was captured, is recorded in a CSV file called `features_metadata.csv`. It also provides file location of the image in the public S3 bucket in the Europe, US East and Asia Pacific regions. A brief description of all fields is presented in Table .

Field name	Description
`chip_id`	Identifier for a single patch
`filename`	Filename of the corresponding image
`satellite`	The source satellite that captured the image
`month`	Month in which the image was collected
`size`	The file size in bytes
`cksum`	A checksum value to verify the data are correct
`s3path_us`	The S3 file location of the image in US East
`s3path_eu`	The S3 file location of the image in Europe
`s3path_as`	The S3 file location of the image in Asia Pacific

Table 3.3: Description of fields in metadata file.[27]

Focusing on ground-truth data, these are provided in a similar way of input ones in terms of correction and size. In fact, for each patch is represented by an image that cover 2,560 meter by 2,560 meter areas at 10 meter resolution (256x256 pixels), with every pixel in the satellite data that corresponds to a pixel in the same position in the LiDAR data for the same chip.

To reach the objective of this thesis, one biomass prediction per chip is generated. In order to consider all the useful information that comes from full-year imagery, it was determined to construct the input in a multi-temporal manner by combining all of the two satellites' bands, for each month, into a single input tensor.

3.2.2 Data preprocessing

In general, in the context of training Deep Neural Networks (DNNs), one of the main challenge that arise is the acceleration and stabilization of training process. In this regard, many techinques has been investigated by researchers over the year, with normalization that is proved very effective in improving the stability, efficiency, and generalization performance of deep neural networks, making them more powerful in a wide range of applications.[28]

Normalization usually entails changing the data in order to adhere to a specific scale or norm: when features have different scales, some features may dominate the learning process, while others may have little influence. By standardizing the data, all features are brought to a common scale, making them equally important during training and mitigate the impact of outliers in the data. Since neural networks deal with optimization algorithms such as gradient descent, normalizing the data definitely leads to faster and more reliable algorithm convergence, as bringing the data to the same scale leads the optimization to not get stuck in local optima. Furthermore, it allows to a better stabilization of the gradient during training with the aim to mitigate the problem of vanishing or exploding gradient.[29]

Specifically to the dataset proposed in this work, where data comes from different data source and can can vary widely among them, normalization can make the models more robust to variations in data, allowing it to handle different types of inputs effectively.

Regarding the features data, the normalization method chosen is Z-score normalization [30]. Z-score is a statistical measurement that describes a value's relationship to the mean of a group of values. The formula to calculate the Z-score for a data point X in a dataset with mean μ and standard deviation σ is:

$$Z = \frac{X-\mu}{\sigma}$$

Standard deviations from the mean are used to quantify the Z-score:

- If a Z-score is 0, it indicates that the data point's score is identical to the mean score.

- A Z-score of 1.0 would indicate a value that is one standard deviation from the mean.

The normalization has been computed separately for each band of the images, by calculating a per-channel mean and per-channel standard deviation. In Table 3.4 are recorded mean and standard deviation values for each band taken in consideration.

Data Source	Band	Min	Max	Mean	Standard Deviation
Sentinel-1	VV asc.	-22.98	-2.25	-11.44	3.17
Sentinel-1	VH asc.	-37.69	-9.93	-18.05	4.36
Sentinel-1	VV desc.	-25.00	0.00	-12.98	5.39
Sentinel-1	VH desc.	-70.00	0.00	-24.13	17.26
Sentinel-2	B2	0.00	11872.00	1632.95	2497.89
Sentinel-2	B3	0.00	11295.00	1614.64	2310.34
Sentinel-2	B4	0.00	11734.00	1604.33	2387.07
Sentinel-2	B5	0.00	12205.00	1922.97	2387.09
Sentinel-2	B6	0.00	12018.00	2486.80	2206.29
Sentinel-2	B7	0.00	11792.00	2598.97	2099.76
Sentinel-2	B8	0.00	12498.00	2746.67	2189.80
Sentinel-2	B8A	0.00	11708.00	2693.65	2025.58
Sentinel-2	B11	0.00	7644.00	1029.67	927.44
Sentinel-2	B12	0.00	6838.00	700.24	753.51
Sentinel-2	CLP	0.00	100.00	12.94	24.59
LiDAR	/	0.00	425.72	63.24	70.89

Table 3.4: Summary of data statistics for Sentinel-1 and Sentinel-2 data sources.

On the other hand, label data has normalized by applying MinMax normalization [31]. In statistics and data processing, MinMax normalization is a method used to rescale numerical variables within a given range, typically between 0 and 1. This kind of feature scaling modifies the data to make it fall inside a given range. The formula for MinMax normalization is:

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

where:

- X is an individual data point
- X_{min} is the minimum value in the dataset for that specific feature.
- X_{max} is the maximum value in the dataset for that specific feature.

Data Source	Min	Max	Mean	Standard Deviation
LiDAR	0.00	425.72	63.24	70.89

Table 3.5: Summary of data statistics for ground-truth data.

Statistics about ground-truth data are summarized in Table 3.5.

Furthermore, to make the training of the chosen models less sensitive to outliers, it was decided to remove them before calculating the statistics and applying preprocessing.

Chapter 4

Methodology

In this chapter the methodology adopted is described, starting from a definition of the problem addressed, followed by a definition of the network architecture employed and loss functions utilized to perform further experiments.

4.1 Problem statement

The work is focused on estimating yearly Aboveground Biomass (AGBM) for Finnish forests using satellite imagery from Sentinel-1 and Sentinel-2 satellite missions.

Specifically, for each patch of forest comprised in the dataset, that represent an area of 2,560 $\times$ 2.560 meter at 10 meter resolution, one biomass prediction per chip is generated. Predictions include a yearly peak AGBM value for each 10 by 10 pixel in the chip.[2]

Since there is a need to predict a value for each pixel of each patch, semantic segmentation, a deep learning technique that predicts a label for each pixel of the input image, was proposed to handle the problem. Having one satellite image for each month covering the period of a full year from two different satellites, it was chosen to aggregate the bands of both satellite images for each month in order to have a multi-temporal and multi-modal input. Consequently, the prediction will consist of a single image that at each pixel has a predicted AGBM value.

4.2 Network architectures

4.2.1 UNet with Attention Pooling

Taking inspiration from the Attention UNet model [14], this work sought a semantic segmentation model that takes into account not only the spatial relations of the

image but also the temporal dimension of the input. In fact, the proposed model enriches an UNet model with an aggregation of the temporal dimension at the encoder level via self-attention module.

The proposed model incorporates the various aspects of “U-shaped" architecture and also adds an attention mechanism, not only to enhance the model's ability to focus on relevant image features during the segmentation process, but also to perform temporal aggregation through a self-attention module. The attention gate is applied to the skip connections between the encoder and decoder paths in the UNet architecture, as exploited in [11]. These connections play a crucial role in information flow, and the attention gate selectively modulates the information passed through these connections. In fact, the attention gate helps the model to selectively focus on important characteristics, improving the segmentation process, by modifying the relevance of features using attention scores. More accurate segmentation and enhanced contextual understanding are made possible by this increased information flow.[32]

In semantic segmentation tasks, it is common practice to use pre-trained neural network architectures as backbones for the encoder in a UNet. Since pre-trained models are trained on large-scale datasets for tasks like image classification, they can actually significantly improve the performance of the UNet model. Examples of pre-trained backbones with proven feature extraction capabilities are VGG [33], ResNet [34], or EfficientNet [35]. These models have deeper layers that capture more abstract and complicated information, while the early layers are skilled at learning low-level features. The pre-trained backbone can be used to extract features in the input image.

After introducing the key points of the model with its strengths, one the of architecture that is proposed to approach the task at hand: it consists of a UNet with attention gates that perform time dimension aggregation via self-attention modules at the encoder level. Specifically, the latter is implemented with a pre-trained backbone of the EfficientNetV2 [35] model.

4.2.2 Swin UNETR

The Swin UNETR (Swin Transformer for UNet based Architecture) [12] is an advanced model that integrates the Swin Transformer [13] with the UNet architecture [14]. It combines the strengths of the UNet's encoder-decoder structure with the powerful self-attention mechanism of the Swin Transformer, resulting in a model tailored for semantic segmentation tasks.[12]

Regarding architectural aspects, the model takes as input $\mathcal{X} \in \mathbb{R}^{H \times W \times D \times S}$ that has a patch resolution (H', W', D'). A sequence of 3D tokens is created by leveraging a patch partition layer, which divide the input image into non-overlapping patches, then they are projected into an embedding space of dimension C. For effective token

interaction modeling, the self-attention is calculated into non-overlapping windows during the partitioning step that are efficiently computed with a 3D-cycling shifting [12] at each encoder layer, with the self-attention performed as:

$$Attention(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d}})V \tag{4.1}$$

where Q, K, V represents respectively query, key and value, whereas d is the size of query and key.

The Swin UNETR encoder comprises 4 stages with 2 transformer blocks at each one. In particular, the first stage layer create $\frac{H}{2} \times \frac{W}{2} \times \frac{D}{2}$ 3D tokens, that are reduced in dimension at each encoder stage by a factor 2 exploiting a patch merging layer. The latter also groups patches and concatenates them resulting in a 4C-dimensional feature embedding, with a subsequent reduction in a 2C embedding space towards a linear layer. In this specific case, C is set to 24. Other hyperapameters, such as the size of the window, are available in Table 4.1.

The decoder follows a more canonical configuration. In fact, at each stage $i \in \{1, 2, 3, 4\}$ the outputs are reshaped into $\frac{H}{2^i} \times \frac{W}{2^i} \times \frac{D}{2^i}$ and then passed to a residual block followed by an instance normalization layer [36]. The resolution is increased by a deconvolutional layer and the output are then concatenated to the outputs of the previous stages and fed to another residual block. At this point of the network, the output has a shape o $H \times W \times D \times$ 24: since we want to predict single yearly AGBM for each pixel, a mean over the third dimension is performed, in order to aggregate the temporal dimension. In the end, the segmentation masks are computed expoliting a sequence of $1 \times 1 \times 1$ convolutional layer followed by a sigmoid activation function. In Figure 4.1 it can be seen the a graphical overview of the implemented model.

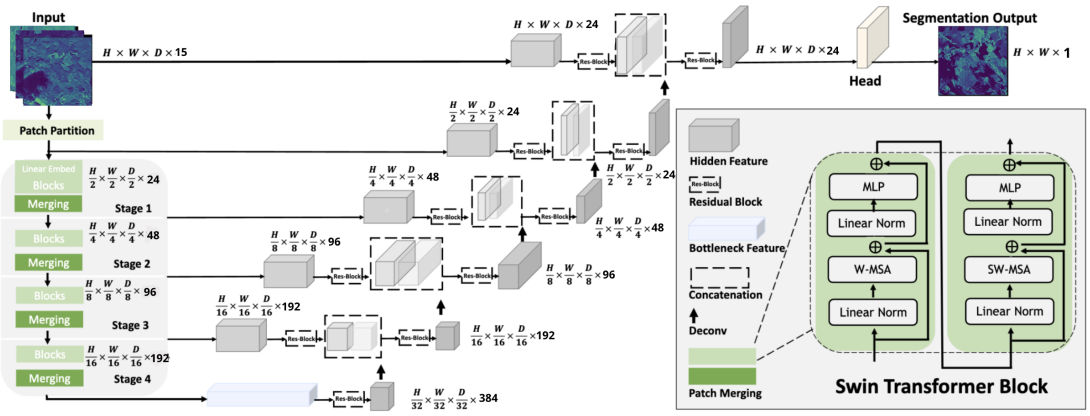

Figure 4.1: Overview of the Swin UNETR model implemented.

Feature Size	Number of Blocks	Window Size	Number of Heads
24	[2,2,2,2]	[3,7,7]	[3,6,12,24]

Table 4.1: Description of Swin UNETR's hyperparameters.

4.3 Loss functions

In this section are presented all the loss function employed in thesis' experiments.

4.3.1 Root Mean Squared Error

In fields like statistics and machine learning, the Root Mean Square Error (RMSE) is a frequently used metric to assess how well a regression model is performing. It provides information on the correctness of the model by calculating the average magnitude of the errors between the actual and projected values.[37]

It is computed as follows:

1. Determine each image pixel's squared error as follows: square the result after deducting the anticipated value from the real (ground truth) value.
2. Compute the mean of the squared errors: Sum all the squared errors and divide by the number of pixels.
3. Calculate the mean squared error's square root: The RMSE value, which is in the same units as the target variable, is what this step provides. The typical amount of the error between the actual and forecasted values is quantified.

Mathematically, the RMSE loss is expressed as:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(y_i - \hat{y}_i)^2} \tag{4.2}$$

- N is the number of pixel.
- y_i represents the actual (ground truth) value.
- $\hat{y}_i$ represents the predicted value.
- The $\sum$ symbol denotes summation over pixels.

Large errors are given more weight by RMSE, which makes it more susceptible to outliers. This indicates that, in comparison to other measures like Mean Absolute Error (MAE), significant prediction mistakes have a greater influence on the RMSE

score.[37] This loss is primarily used in regression tasks, where the model aims to predict a continuous target variable. Although this work does not deal with a regression task, still there is a need to predict a continuous target for each pixel in the image, and this is the reason why some experiments involving this loss function.

4.3.2 Mean Absolute Error

In the problems of the same kind that we are addressing in this work, it is often used the Mean Absolute Error (MAE) [38] metric to calculate the average magnitude of errors between the predicted and actual values. It offers a simple and natural understanding of the model's functionality. The MAE is calculated by taking the average of the absolute differences between predicted values and true values. Mathematically, for N data points, that in this specific case corresponds to image's pixels:

$$MAE = \frac{1}{N}\sum_{i=0}^{N}|y_i - \hat{y}_i| \tag{4.3}$$

Where:

- y_i represents the true value of the target variable for the i-th pixel.
- $\hat{y}_i$ represents the predicted value of the target variable for the i-th pixel.
- N represents the total number of pixels.

4.3.3 Structural Similarity Index

A popular metric in computer vision and image processing for determining how similar two images are to one another is the Structural Similarity Index (SSIM) [39]. It is used to evaluate the perceptual differences between images, accounting for structural information, brightness, contrast, and differences in pixels. The formula for the Structural Similarity Index (SSIM) is the follow:

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \tag{4.4}$$

Where:

- x and y are the two images being compared.
- μ_x and μ_y are the means of x and y, respectively.
- σ_x and σ_y are the standard deviations of x and y, respectively.
- σ_{xy} is the covariance between x and y.

- C_1 and C_2 are small constants added for numerical stability.

The SSIM value is a number between -1 and 1, where 1 denotes perfect similarity and 0 denotes no similarity at all between the images. Greater similarity between the images is indicated by higher SSIM values, which take into account both local and global image properties. [40]

4.3.4 Focal Frequency Loss

In Chapter 2.4, Focal Frequency Loss [22] has been presented. It aims to measure the gap related to frequency domain between predictions and ground-truth, by allowing a model to adaptively focus on frequency components that are hard to synthesize by down-weighting the easy ones.

The loss function is introduced mainly for reconstruction and synthesis tasks for generative models in order to mitigate the effects of *spectral bias* [21], a learning bias of neural networks towards low-frequency functions. Consequently, generative models have a tendency to converge to an inferior point by es-chewing hard frequencies, or frequency components that are difficult to synthesis. Specifically to this work, we want to analyze the frequency gap between the predictions and the ground-truth values, and this can be exploited by transforming both labels and predictions to their frequency representations using the standard discrete Fourier transform (DFT). The 2D discrete Fourier transform is used to translate an image into its frequency representation:

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{i2\pi(\frac{ux}{M}+\frac{vy}{N})} \tag{4.5}$$

Where:

- The image size is $M \times N$
- (x, y) denotes the coordinate of an image pixel in the spatial domain
- $f(x, y)$ is the pixel value
- (u, v) represents the coordinate of a spatial frequency on the frequency spectrum
- $F(u, v)$ is the complex frequency value; e and i are Euler's number and the imaginary unit, respectively.

Let $F_p(u, v)$ be the spatial frequency value at the spectrum coordinate (u, v) of the models' predicted mask, and the corresponding $F_{gt}(u, v)$ with the similar meaning

w.r.t. the ground-truth value, the frequency distance between the prediction and ground truth images can be written as the average value over all image pixels:

$$d(F_p, F_{gt}) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} |F_p(u,v) - F_{gt}(u,v)|^2 \tag{4.6}$$

However, as stated in [22], since each frequency has the same weight, using this equation directly as a loss function is ineffective for handling hard frequencies. To overcome this issue, a spectrum weight matrix $w(u,v)$ is created to down-weight the easier-to-learn frequencies. During training, a non-uniform distribution on the current loss of each frequency determines the spectrum weight matrix dynamically. Every image has a unique matrix of spectrum weights. The matrix and the spectrum share the same shape. By performing the Hadamard product for the spectrum weight matrix and the frequency distance matrix, it is defined the full form of the focal frequency loss (FFL):

$$d(F_p, F_{gt}) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} w(u,v)|F_p(u,v) - F_{gt}(u,v)|^2 \tag{4.7}$$

Chapter 5

Experiments

In this chapter it is provided an overview of the configurations adopted to train and test the model, with an introduction to the evaluation metric chosen to assess the goodness of results. The results of all the experiments are then presented and discussed.

5.1 Implementation details

This section presents all the implementation choices in order to train and test the models involved in the thesis work. In particular, the selection of hyperparameters is introduced alongside the split of the dataset in training, validation and test set.

5.1.1 Data preparation

The two models have been trained considering a multi-temporal input by aggregating the spectral bands of images from Sentinel-1 and Sentinel-2. In fact, the networks receive an input tensor of size $12 \times 15 \times 256 \times 256$ (this is the case of the UNet model with attention pooling) or $15 \times 12 \times 256 \times 256$ (as in the case of the Swin UNETR [12]), where 12 represents the number of months for which an image is available, 15 is the total number of spectrum bands (both from Sentinel-1 and Sentinel-2), and 256x256 is the size of each satellite image.

As mentioned earlier in Chapter 3, the dataset contains about 8,689 patches (or chips) of forest and was initially divided into training and test sets. The idea is to perform a K-Fold cross validation using $K = 5$ folds. This results in training five different models and the prediction is generated for each of them. The final prediction, which will be the subject of the chosen evaluation metric, will be obtained by averaging the results from the five models. For this reason, five folds were generated from the training set, and for each training four of them represent

the input data of the networks and the rest are considered as validation sets. In Table 5.1 a summary of the dataset's split is presented.

	Training	Validation	Test	Total
Number of samples	5560	1391	1738	8689

Table 5.1: Sample distribution for training, validation, and test sets.

5.1.2 Scenarios definition

In performing the experiments in this thesis, different configurations were devised in order to explore different ways of training the models and also to evaluate which type of loss function performs best with respect to the objective of the work, trying to make sure that we have an efficient and fast training process.

In this sense, two different scenarios of experiments are introduced, involving the two different models, both trained with various types of loss for a number of epochs to allow for a reasonable training time.

Specifically, Scenario 1 involves experiment regarding UNet with Attention Pooling model, which is trained by minimizing the RMSE Loss [37] in a first configuration, and then with the Focal Frequency Loss [22]. In the former, it's important to highlight that the loss has been computed only for pixels that have AGBM values under a certain threshold (400 in this case), in order to make the training less sensitive to predictions' outliers. For both experiments, 200 has been set as number of epochs for training.

In Scenario 2, which comprises Swin UNETR's training, the first loss function considered to minimize is a weighted sum of two aforementioned functions presented in Chapter 4.3.2 and 4.3.3: MAE [38] and SSIM [39]. In particular, the loss is formulated as follows:

$$L_s = L_{MAE} + 0.2 \cdot L_{SSIM} \tag{5.1}$$

Where:

- L_{MAE} represents the MAE loss value
- L_{SSIM} represents the SSIM

Other type of experiment in this scenario takes into account the minimization of Focal Frequency Loss. In this case the network has been trained for 100 epochs.

In Table 5.2 is presented a summary of different scenarios devised.

Name	Model	Losses	Epochs
Scenario 1	UNet with Attention Pooling	a) RMSE Loss b) Focal Frequency Loss	200
Scenario 2	Swin UNETR	a) MAE + SSIM b) Focal Frequency Loss	100

Table 5.2: Summary of experiments' scenarios.

5.1.3 Multi-GPUs training setup

Training a deep learning model often requires a lot of computational resources and may take a long time. Indeed, in this case, training the model on a single GPU makes the problem intractable. To mitigate the problem, a multi-GPU strategy was adopted to make the training process easier and faster.

Specifically to this work, Distributed Data Parallel (DDP) [41] is implemented. DDP is a deep learning approach that trains neural networks over several GPUs or computing devices in a distributed computing environment.By distributing the effort among several devices, such as individual GPUs inside a single computer or several machines in a cluster, it permits the concurrent training of a model. DDP implements data parallelism, allowing each GPU to work with a subset of the training set. Devices exchange information in order to synchronize the model's parameters and share gradients. To update the weights of the model and aggregate gradients across devices, techniques like AllReduce operations are frequently employed.[42] This method works especially well for models that deal with big datasets or demand a lot of processing power. Furthermore, DDP may improve distributed training setups' fault tolerance. To minimize interruption in the event of a single machine or device failure, the training process can be carried out on additional functional devices.

In the experiments concerning Scenario 1, DDP is implemented via python leveraging PyTorch framework in order to train the network over 2 GPUs at hand. This implementation exploits replication of the models on all the devices; each replica calculates gradients and simultaneously synchronizes with the others using the ring AllReduce algorithm.

5.1.4 Training process

The process of training a neural network is often computationally heavy and very time-consuming, which is why it is tried to carefully choose the various components that are involved in the process to make the training easier in all respects. In fact, the main players in the process are the loss function optimizer that is sought to

be minimized, the learning rate and the data augmentation techniques necessary to make the model more accurate in its prediction. When training deep neural networks, it is often useful to reduce learning rate as the training progresses. This can be done by using pre-defined learning rate scheduler, that attempt to regulate the learning rate during training by lowering the learning rate on a predetermined schedule.[43] In this work, the Cosine Annealing Learning Rate Scheduler (firstly introduced in [44]) is employed, that it is a form of learning rate schedule: it starts with a high learning rate and swiftly decreasing to a low number before rapidly increasing again. An high value of this hyperparameter keep the learner to getting stuck in a local cost minima; a low value is then reached to allow it to converge to a near-true-optimal point within the global minima it finds. In the schedule a warm restart strategy is exploited, that involves periodically resetting of learning rate to inital value after a certain number of iterations.[45] The learning rate at iteration t is calculated according to this formulation:

$$\eta_t = \eta_{min} + \frac{1}{2}(\eta_{max} - \eta_{min})(1 + \cos(\frac{T_{cur}}{T_i}\pi)) \tag{5.2}$$

Where:

- T_{cur} is the number of epochs since last restart
- T_i is the number of epochs between two warm restarts
- η_{max} represents the maximum value that learning rate can assume (the initial value)
- η_{min} represents the minimum value that learning rate can assume

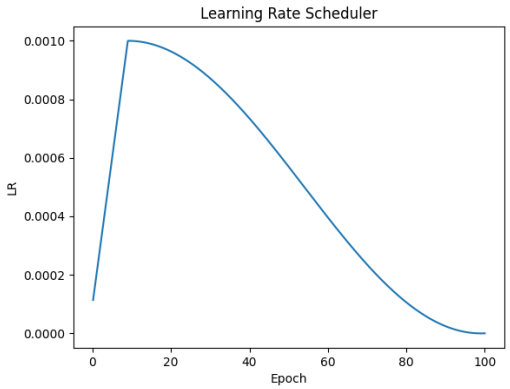

Figure 5.1: Example of learning rate schedule employed in Scenario 2 experiments.

It's important to note that when $T_{cur} = T_i$, η_t is equal to η_{min}. Instead, when a restart occurs ($T_{cur} = 0$), $\eta_t = \eta_{max}$.[45] In Figure 5.1 is highlighted the learning rate behavior according to the schedule employed.

Architecture definitions, network training method implementation were implemented through the python language using the PyTorch Lightning framework, while data preprocessing techniques were developed using the numpy, scikit-learn and pandas libraries. Data augmentation (which will be introduced later), on the other hand, was done through a package called "volumentations" for 3D Volume data augmentation. Through the Table 5.3, an overview of the versions of the corresponding python packages is provided.

Package	**Version**
efficientnet-pytorch	0.7.1
focal-frequency-loss	0.3.0
keras	2.14.1
matplotlib	3.8.0
monai	1.2.0
numpy	1.24.3
pandas	2.0.1
piqa	1.3.1
pytorch-lightning	2.0.8
rasterio	1.3.6
scikit-learn	1.2.2
segmentation-models-pytorch	0.3.2
tensorboard	2.14.1
tifffile	2023.4.12
timm	0.6.12
torch	1.13.1
torchvision	0.14.1
volumentations-3D	1.0.4

Table 5.3: Package list with associated version.

The learning rate is an hyperparameter that indicates how our network's weights are adjusted in relation to the loss gradient descent. It establishes the pace at which we will approach the ideal weights. The learner shall forgo finding the optimal solution if the learning rate is really high. Conversely, it will require an excessive number of iterations to reach the optimal values if learning rate value is too little. Thus, it is essential to use a good value for it.[46]

The optimizer employed in the experiments is AdamW [47], which is an extension of Adam [48]. Adam is an optimization algorithm that may be used to update

network weights iteratively based on training data, in place of the traditional stochastic gradient descent process. It follows the idea of maintainining and adapt learning rate for each network parameter, in contrast to what is done with the more classical stochastic gradient descent, where a single learning rate is manteined for all weight updates and it doesn't change during training. Adam uses the average of the second moments of the gradients (the uncentered variance) instead of only modifying the parameter learning rates based on the average first moment (the mean). The procedure computes an exponential moving average of both the gradient and the squared gradient. The moving averages' decay rates are regulated by the constants β_1 and β_2.[49] AdamW provide a different weight decay handling in order to not create interference with the adaptive learning rate behavior. Weight decay is a regularization term that penalizes large weights in the network to prevent overfitting, and AdamW correctly decouples the weight decay terms update from the optimization steps.

Data augmentation techinques are crucial in order to help create a more representative dataset and increase its diversity: this leads to an increasing of the generalization and robustness of the model. In this work, a 3D Volume data augmentation is considered through *volumentations-3D* [50] python library. The transofrmation applied to data are horizontal and vertical flip and 90 degree rotation, each one with probability to apply the transformation $p = 0.1$.

A summary of hyperparameters' values employed in the two scenarios is presented in Table 5.4.

The experiments concerning Scenario 1 were conducted using a workstation equipped with 2 Nvidia A100 GPUs with 80GB of RAM, with each experiment, which involves training the network for 200 epochs, taking about 4 days. On the other hand, for those described in Scenario 2, the Legion Cluster provided by HPC@POLITO [51], and each experiment, that comprises network training for 100 epochs, takes about 2 days to complete.

Parameter	**Value**
Batch size	8
Optimizer	AdamW
Learning rate (LR)	0.001
Weight decay	0.01
LR Scheduler	Cosine Annealing

Table 5.4: Summary of hyperparameters' values.

5.2 Evaluation metric

The Average Root Mean Square Error (RMSE) [37] is a statistic used to assess the performance of the model. RMSE is the square root of the mean of squared differences between estimated and observed values. The Root Mean Square Error (RMSE) will be computed for each patch pixel by comparing it to its corresponding pixel in the ground-truth. Every image in the test set will have its RMSE computed and then averaged. Since this is an error metric, a smaller number is preferable. Although there are some outliers in data, they are included in the scoring, and also pixels with value of zero.[2]

$$AverageRMSE = \frac{\sum_{i=0}^{M} \sqrt{\frac{1}{N} \sum_{i=0}^{N} (y_i - \hat{y}_i)^2}}{M} \tag{5.3}$$

Where:

- M is the number of patches in the dataset.
- N is the number of pixel of the image.
- y_i represents the actual (ground truth) value.
- $\hat{y}_i$ represents the predicted value.

5.3 Results

Based on the selected evaluation metric that was presented in the previous subsection, the results for the various experiment scenarios are categorized and examined in this section. Overall, the examined models produce outcomes that are comparable with the winners of "The BioMassters" competition and significantly outperform the baseline algorithm supplied by the competition's organizers, which involved using a UNet model that reach an evaluation metric of 101.98. The results of the experiments carried out in this work are summarized in Table 5.5.

What comes to light is that the experiments defined in Scenario 2, specifically involving the training of the Swin UNETR model, achieve better results than the UNet model with attention pooling. In fact, analyzing the table, it can be seen that each fold of the first model returns better performance in terms of evaluation metrics, which is also highlighted when the model ensemble of the 5 generated folds is performed. In fact, in this case, Swin UNETR reaches 27.33, compared to the model in Scenario 1, which reaches a value of 29.95. The reason probably lies in the fact that the model involved in Scenario 2, besides being more robust and recent, has a self-attention mechanism based on partitioning the image into

Scenario	Loss	Fold 0	Fold 1	Fold 2	Fold 3	Fold 4	Losses
Scenario 1	RMSE	29.05	30.14	29.77	29.84	29.71	29.95
Scenario 1	FFL	31.18	32.60	31.65	31.45	31.79	31.54
Scenario 2	MAE+SSIM	**27.40**	**27.25**	**27.37**	**27.35**	**27.27**	**27.33**
Scenario 2	FFL	30.30	30.18	30.40	30.11	30.25	30.25

Table 5.5: Performance metrics for different experiment scenarios and folds in terms of RMSE.

non-overlapping windows that is probably more efficient than the aggregative attention module implemented in UNet with attention pooling.

The results are also of the same type when the experiments of Scenario 1 and Scenario 2 involving the same loss function, i.e. Focal Frequency Loss, are compared. In this regard, in fact, the models have the same behavior both when the inference is performed for each individual fold and when the model ensemble is run: regarding the latter aspect, Swin UNETR is able to achieve a value of 30.25 of RMSE, which is better than that achieved by UNet with attention pooling (31.54).

Analyzing the results obtained in more detail, placing the focus on the experiments involving the FFL, we highlight the fact that these, for both scenarios, produced results that in general achieved a better RMSE than the baseline model, but nevertheless are worse than the other configurations of experiments involving minimization of other loss functions used in this work. If we analyze the numerical data provided by the Table 5.5, in fact, it can be seen that the training of the SwinUNETR with FFL achieves an evaluation metric of 30.25 with model ensembling while its counterpart trained with the loss composed of MAE + SSIM achieves 27.33. An outcome that reflects the same behavior is also shown in the experiments of Scenario 1, where the UNet with attention pooling reaches 31.54 when trained with FFL, while with RMSE Loss it is able to achieve a better outcome in terms of evaluation metric, which is 29.95. Although there is a subtle difference in terms of evaluation metric that indicates that the use of this loss does not improve the performance of the models employed, the segmentation mask produced is still quite comparable with that coming from the models trained with other configurations. In this regard, a comparison between ground-truth and output of the SwinUNETR trained with and without FFL can be seen in Figure 5.2.

The frequency spectra produced by both ground-truth and prediction made by the models were qualitatively analyzed to investigate why tranining with this loss was not so effective for the models that were chosen and implemented for the experiments. Graphical examples are shown in Figure 5.3 to make a comparison of the prediction and their corresponding ground truth for some forest patches that were used as input data, providing also the frequency spectra of both. What is

highlighted is that the frequency spectra in general are affected by significant noise, which do not allow all image frequencies to be encoded correctly and perform pixel-per-pixel distance minimization in the frequency domain. Through this analysis, it could therefore be found that the main reason why traning with this loss function in general does not produce strong results would be due to the excessive noise of the frequency spectra.

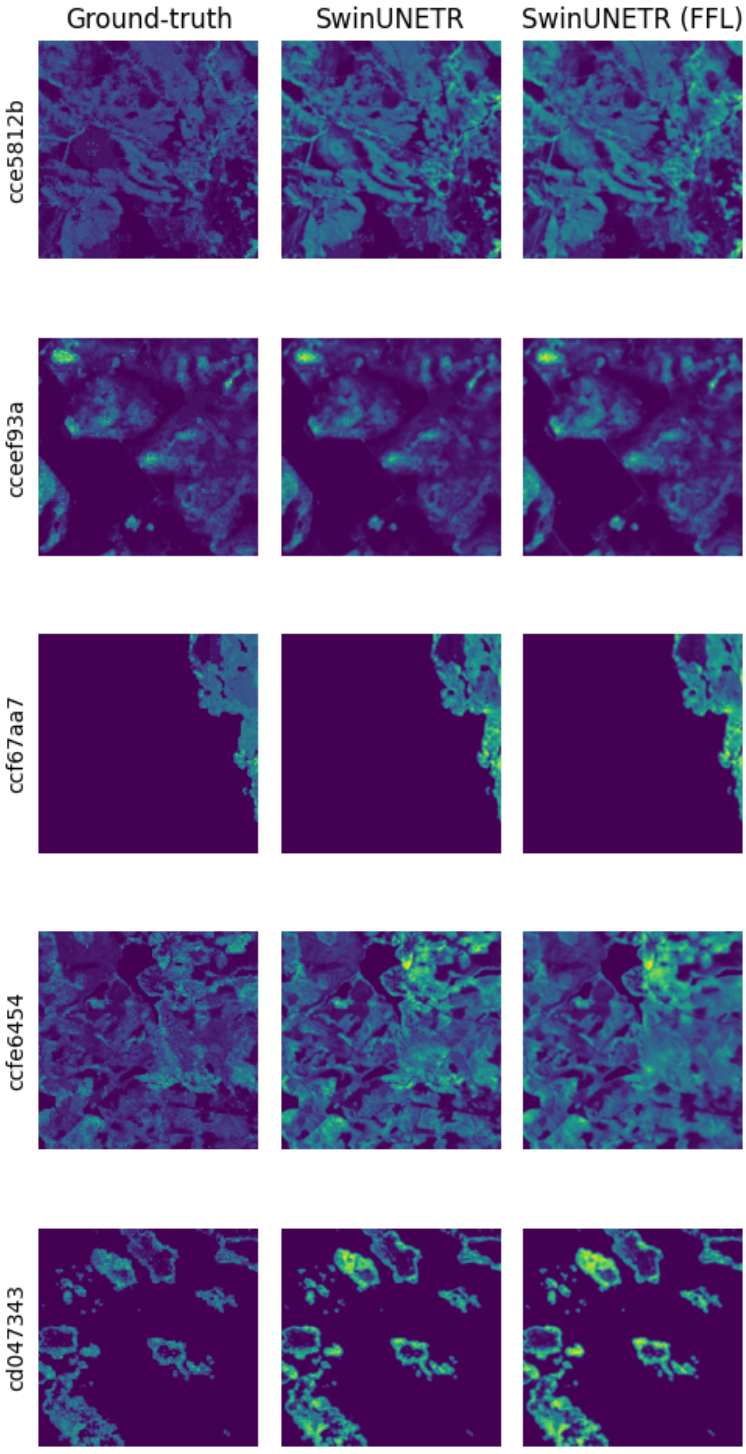

Figure 5.2: Comparison between ground-truth data and Swin UNETR's outputs.

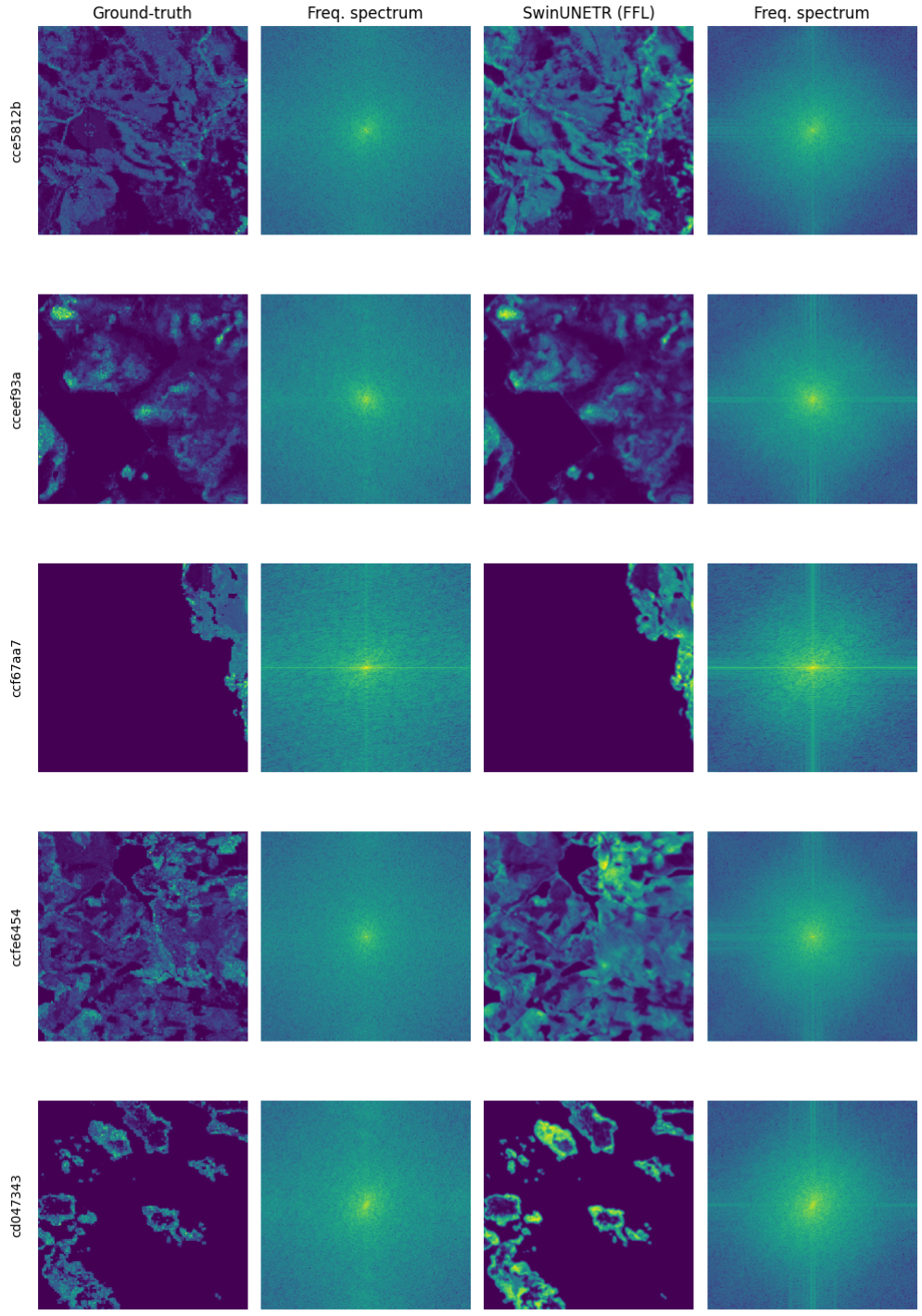

Figure 5.3: Comparison between frequency spectra of ground truth and Swin UNETR's outputs.

Chapter 6

Conclusions

The main purpose of the thesis work focused on exploring deep learning techniques to provide annual Aboveground Biomass (AGBM) estimation for patches of Finnish forests. To this end, two models that represent state-of-the-art for semantic segmentation tasks were implemented and evaluated. Specifically, by processing satellite imagery from the Sentinel-1 and Sentinel-2 space missions, combined with ground-truths measured by airborne LiDAR, a segmentation mask was determined by starting with multi-temporal input and aggregating the information from the two satellites. The experiments involved different training configurations of the models, shows that the presented algorithms significantly improved the baseline that had been provided by the organizers of "The BioMassters" competition, that aims to solve the same task, achieving results very close and comparable to those of the winners, and in some cases even outperforming them.

Specifically, the results highlight how the Transformer-based SwinUNETR model adopted in the work succeeds in achieving robust results in terms of RMSE, resulting in a validated candidate to provide an alternative remote sensing method to those already used in forest conservation analysis.

On the other hand, the simpler model such as UNet with Attention Pooling still offers good results but with certainly room for improvement. For example, one could experiment whether longer training, in terms of epochs, could lead to an improvement of the results already obtained in the thesis.

Furthermore, at the time this work is being done, the test set on which the results of those who proposed a valid solution to the competition participation were evaluated has not been released. This is why the available data were split into training, validation and test set in order to properly evaluate proposed models. However, the work presented could have a continuation in performing training of the models with a much larger training set to obtain better performance.

Bibliography

[1] UNECE. *Carbon Sinks and Sequestration*. URL: https://unece.org/forests/carbon-sinks-and-sequestration (cit. on p. 1).

[2] DrivenData. *Overview of "The Biomassters competition"*. URL: https://www.drivendata.org/competitions/99/biomass-estimation/page/534/ (cit. on pp. 1, 2, 12, 13, 21, 34).

[3] «Methods for Estimating Above-Ground Biomass». In: *Carbon Inventory Methods Handbook for Greenhouse Gas Inventory, Carbon Mitigation and Roundwood Production Projects*. Dordrecht: Springer Netherlands, 2008, pp. 113–147. ISBN: 978-1-4020-6547-7. DOI: 10.1007/978-1-4020-6547-7_10. URL: https://doi.org/10.1007/978-1-4020-6547-7_10 (cit. on p. 1).

[4] Trinh Huynh, David J. Lee, Grahame Applegate, and Tom Lewis. «Field methods for above and belowground biomass estimation in plantation forests». In: *MethodsX* 8 (2021), p. 101192. ISSN: 2215-0161. DOI: https://doi.org/10.1016/j.mex.2020.101192. URL: https://www.sciencedirect.com/science/article/pii/S221501612030412X (cit. on p. 2).

[5] NOAA - National Oceanic and Atmospheric Administration. *What is LiDAR?* URL: https://oceanservice.noaa.gov/facts/lidar.html (cit. on pp. 2, 16).

[6] ESA - European Space Agency. *Sentinel-1*. URL: https://sentinel.esa.int/web/sentinel/missions/sentinel-1 (cit. on pp. 2, 12).

[7] ESA - European Space Agency. *Sentinel-2*. URL: https://sentinel.esa.int/web/sentinel/missions/sentinel-2 (cit. on pp. 2, 14).

[8] ESA - European Space Agency. *SAR Instrument*. URL: https://sentinels.copernicus.eu/web/sentinel/technical-guides/sentinel-1-sar/sar-instrument (cit. on p. 2).

[9] ESA - European Space Agency. *MultiSpectral Instrument (MSI) Overview*. URL: https://sentinels.copernicus.eu/web/sentinel/technical-guides/sentinel-2-msi/msi-instrument (cit. on p. 2).

[10] DrivenData. *DrivenData presentation*. URL: https://www.drivendata.org (cit. on pp. 2, 12).

[11] Ozan Oktay et al. *Attention U-Net: Learning Where to Look for the Pancreas*. 2018. arXiv: 1804.03999 [cs.CV] (cit. on pp. 2, 4, 6, 22).

[12] Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, Dong Yang, Holger Roth, and Daguang Xu. *Swin UNETR: Swin Transformers for Semantic Segmentation of Brain Tumors in MRI Images*. 2022. arXiv: 2201.01266 [eess.IV] (cit. on pp. 2, 4, 8, 9, 22, 23, 28).

[13] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. 2021. arXiv: 2103.14030 [cs.CV] (cit. on pp. 2, 7, 8, 22).

[14] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV] (cit. on pp. 4–6, 8, 21, 22).

[15] Premanand S. *A Comprehensive Guide to UNET Architecture*. URL: https://www.analyticsvidhya.com/blog/2023/08/unet-architecture-mastering-image-segmentation (cit. on pp. 4, 5).

[16] Javier Fernandez. *Attention in Computer Vision*. URL: https://towardsdatascience.com/attention-in-computer-vision-fd289a5bd7ad (cit. on p. 6).

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL] (cit. on p. 6).

[18] Chinmay Bhalerao. *Vision Transformers [ViT]: a very basic introduction*. URL: https://medium.com/data-and-beyond/vision-transformers-vit-a-very-basic-introduction-6cd29a7e56f3 (cit. on pp. 7, 8).

[19] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV] (cit. on p. 8).

[20] Ali Hatamizadeh, Yucheng Tang, Vishwesh Nath, Dong Yang, Andriy Myronenko, Bennett Landman, Holger Roth, and Daguang Xu. *UNETR: Transformers for 3D Medical Image Segmentation*. 2021. arXiv: 2103.10504 [eess.IV] (cit. on p. 8).

[21] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. *On the Spectral Bias of Neural Networks*. 2019. arXiv: 1806.08734 [stat.ML] (cit. on pp. 9, 26).

[22] Liming Jiang, Bo Dai, Wayne Wu, and Chen Change Loy. *Focal Frequency Loss for Image Reconstruction and Synthesis*. 2021. arXiv: 2012.12821 [cs.CV] (cit. on pp. 9–11, 26, 27, 29).

[23] *European Space Agency*. URL: https://www.esa.int/ (cit. on p. 12).

[24] *Copernicus programme*. URL: https://www.copernicus.eu/en/about-copernicus (cit. on p. 12).

[25] ESA - European Space Agency. *Sentinel-1 SAR Acquisition Modes*. URL: https://sentinels.copernicus.eu/web/sentinel/user-guides/sentinel-1-sar/acquisition-modes (cit. on p. 13).

[26] Sentinel Hub. *Sentinel-1 GRD Data*. URL: https://docs.sentinel-hub.com/api/latest/data/sentinel-1-grd/ (cit. on p. 13).

[27] DrivenData. *"The BioMassters" competition problem description*. URL: https://www.drivendata.org/competitions/99/biomass-estimation/page/536/ (cit. on pp. 13–17).

[28] Maciej Balawejder. *Overview of Normalization Techniques in Deep Learning*. URL: https://medium.com/nerd-for-tech/overview-of-normalization-techniques-in-deep-learning-e12a79060daf (cit. on p. 18).

[29] Zhe Ming Chng. *Using Normalization Layers to Improve Deep Learning Models*. URL: https://machinelearningmastery.com/using-normalization-layers-to-improve-deep-learning-models/ (cit. on p. 18).

[30] Mohammed Z Al-Faiz, Ali A Ibrahim, and Sarmad M Hadi. «The effect of Z-Score standardization (normalization) on binary input due the speed of learning in back-propagation neural network». In: *Iraqi Journal of Information and Communication Technology* 1.3 (2018), pp. 42–48 (cit. on p. 18).

[31] S. Gopal Krishna Patro and Kishore Kumar Sahu. *Normalization: A Preprocessing Stage*. 2015. arXiv: 1503.06462 [cs.OH] (cit. on p. 19).

[32] Robin Vinod. *A detailed explanation of the Attention U-Net*. URL: https://towardsdatascience.com/a-detailed-explanation-of-the-attention-u-net-b371a5590831 (cit. on p. 22).

[33] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV] (cit. on p. 22).

[34] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV] (cit. on p. 22).

[35] Mingxing Tan and Quoc V. Le. *EfficientNetV2: Smaller Models and Faster Training*. 2021. arXiv: 2104.00298 [cs.CV] (cit. on p. 22).

[36] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. *Instance Normalization: The Missing Ingredient for Fast Stylization*. 2017. arXiv: 1607.08022 [cs.CV] (cit. on p. 23).

[37] Wikipedia contributors. *Root-mean-square deviation — Wikipedia, The Free Encyclopedia*. [Online; accessed 17-November-2023]. 2023. URL: https://en.wikipedia.org/w/index.php?title=Root-mean-square_deviation&oldid=1179174918 (cit. on pp. 24, 25, 29, 34).

[38] Wikipedia contributors. *Mean absolute error — Wikipedia, The Free Encyclopedia*. https://en.wikipedia.org/w/index.php?title=Mean_absolute_error&oldid=1171541586. [Online; accessed 17-November-2023]. 2023 (cit. on pp. 25, 29).

[39] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. «Image quality assessment: from error visibility to structural similarity». In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: 10.1109/TIP.2003.819861 (cit. on pp. 25, 29).

[40] Pranjal Datta. *All about Structural Similarity Index (SSIM)*. URL: https://medium.com/srm-mic/all-about-structural-similarity-index-ssim-theory-code-in-pytorch-6551b455541e (cit. on p. 26).

[41] Shen Li et al. *PyTorch Distributed: Experiences on Accelerating Data Parallel Training*. 2020. arXiv: 2006.15704 [cs.DC] (cit. on p. 30).

[42] Suraj Subramanian. *What is Distributed Data Parallel (DDP)*. URL: https://pytorch.org/tutorials/beginner/ddp_series_theory.html (cit. on p. 30).

[43] Suki Lau. *Learning Rate Schedules and Adaptive Learning Rate Methods for Deep Learning*. URL: https://towardsdatascience.com/learning-rate-schedules-and-adaptive-learning-rate-methods-for-deep-learning-2c8f433990d1 (cit. on p. 31).

[44] Ilya Loshchilov and Frank Hutter. *SGDR: Stochastic Gradient Descent with Warm Restarts*. 2017. arXiv: 1608.03983 [cs.LG] (cit. on p. 31).

[45] Hasty.ai. *Cosine Annealing Learning Rate Scheduler*. URL: https://wiki.cloudfactory.com/docs/mp-wiki/scheduler/cosineannealinglr (cit. on pp. 31, 32).

[46] Jason Brownlee. *Understand the Impact of Learning Rate on Neural Network Performance*. URL: https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/ (cit. on p. 32).

[47] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG] (cit. on p. 32).

[48] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG] (cit. on p. 32).

[49] Jason Brownlee. *Gentle Introduction to the Adam Optimization Algorithm for Deep Learning*. URL: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/ (cit. on p. 33).

[50] ZTurbo. *Volumentations-3D Overview*. URL: https://github.com/ZFTurbo/volumentations (cit. on p. 33).

[51] HPCPoliTO. *HPCPoliTO*. URL: http://www.hpc.polito.it (cit. on p. 33).