INSTITUTO
UNIVERSITÁRIO
DE LISBOA

Land cover automatic classification using deep learning techniques applied to satellite imagery

Sérgio Filipe Paiva da Silva Gonçalves dos Santos

Masters in Telecommunications and Computer Engineering

Supervisor:
Prof. Doctor Luís Miguel Martins Nunes, Associate Professor,
ISCTE-IUL

Co-Supervisor:
Prof. Doctor Tomás Gomes da Silva Serpa Brandão, Assistant Professor,
ISCTE-IUL

November, 2023

TECNOLOGIAS
E ARQUITETURA

Department of Information Science and Technology

Land cover automatic classification using deep learning techniques applied to satellite imagery

Sérgio Filipe Paiva da Silva Gonçalves dos Santos

Masters in Telecommunications and Computer Engineering

Supervisor:
Prof. Doctor Luís Miguel Martins Nunes, Associate Professor,
ISCTE-IUL

Co-Supervisor:
Prof. Doctor Tomás Gomes da Silva Serpa Brandão, Assistant Professor,
ISCTE-IUL

November, 2023

Acknowledgments

I want to start the acknowledgments by expressing my gratitude towards my mentors, Luis Nunes and Tomás Brandão. Their patience, along with the help and insights provided, was invaluable throughout this work.

These thanks can be extended to ISCTE-IUL for providing me with opportunities that made me grow and helped me arrive at this moment.

I am also thankful to the DGT team for their willingness to meet with me and provide the data needed for the completion of this dissertation.

I want to express my gratitude to my friends, as their assistance made this task more manageable. Your support and presence were invaluable throughout this whole path, and your friendship means a lot to me.

Lastly, I would like to extend a heartfelt and special thank you to my parents and family. You have always been there for me throughout my life, and during this thesis, it was once again proven how crucial you are in my life. I cannot adequately express how grateful I am for your patience, help, and unwavering support, as well as the values you have instilled in me. At this special moment in my life and the closing of this cycle, I also want to remember and honor my grandparents. I miss them dearly. I am confident that they would be incredibly proud of me now.

This dissertation is also the result of everyone who has assisted the way.

This research task was partially supported by Fundação para a Ciência e a Tecnologia, I.P. (FCT) [ISTAR Projects: UIDB/04466/2020 and UIDP/04466/2020].

Abstract

In this era of population growth and rapid urbanization, effective and sustainable urban development is a very important factor. In this context, Machine Learning (ML) can play a leading role in helping with these tasks; it makes possible the treatment of remote sensing images in a shorter time frame. This thesis focuses on the development and use of Convolutional Neuronal Network (CNN)s to handle multispectral images.

The principal goal is to evaluate the performance metrics and computational complexity of a CNN-based land cover classification approach. And to try and assess if the results achieved are better or worse than the architectures currently implemented by the Direção Geral do Território (DGT). In this order, it was first necessary to understand the provided data and all its inherent characteristics. This data was then preprocessed, and the architecture was defined.

The results show that CNNs present a promising alternative in this context to the implemented methods for land cover classification. Despite the promise it provides, it also highlights the difficulties faced and how the work can be improved, specifically concerning the lack of labeled data. The existence of these difficulties presents opportunities for further development of this work.

As an overview of this dissertation, it is possible to say that the investigation into the feasibility of using CNNs for land cover classification provided positive results. There is, however, as would be expected, room for improvement, especially in what concerns the pre-processing of data.

Keywords: Multispectral imaging, Convolutional Neural Networks, Machine Learning, Land Use Land Cover Classification, Sentinel-2.

Resumo

Nesta em que vivemos de crescimento populacional e rápida urbanização, o desenvolvimento urbano eficaz e sustentável é um fator muito importante. Neste contexto, a aprendizagem automática pode desempenhar um papel fundamental na realização destas tarefas; A utilização deste tipo de algoritmos possibilita por exemplo o tratamento de vários tipos de imagens rapidamente. Esta tese centra-se no desenvolvimento e uso de Redes Convolucionais Neuronais para lidar com imagens multiespectrais.

O principal objetivo da tese é avaliar a taxa de acertos e a complexidade computacional de uma abordagem de classificação da cobertura do solo baseada em redes neuronais convolucionais. Além disto, tentar avaliar se os resultados alcançados são melhores ou piores do que as atuais soluções da DGT. Neste sentido, primeiro foi necessário compreender os dados fornecidos e todas as suas características inerentes. Esses dados foram então pré-processados, e a arquitetura definida.

Os resultados mostram que as CNNs apresentam uma alternativa promissora no âmbito da classificação da cobertura do solo. Apesar da promessa que oferece, esta dissertação também destaca as dificuldades enfrentadas e como o trabalho pode ser melhorado, especificamente no que diz respeito à falta de dados etiquetados. A existência destas dificuldades oferece oportunidades para um desenvolvimento futuro deste trabalho.

Em resumo, acerca desta dissertação, é possível dizer que que a viabilidade da utilização de CNNs para classificação da cobertura do solo foi provada, contudo, como seria de esperar, existe ainda margem para melhorias. Estas melhorias podem recair especialmente por exemplo no contexto do pré-processamento dos dados.

Palavras-chave: Imagens multispetrais, Redes Neuronais convolucionais, Aprendizagem Automática, Classificação da Superficie Terrestre, Sentinel-2.

Contents

List of Figures

List of Tables

Acronyms

AI: Artificial Inteligence.
ANN: Artificial Neuronal Network.

CNN: Convolutional Neuronal Network.

DGT: Direção Geral do Território.
DSR: Design Science Research.

ESA: European Space Agency.

GPU: Graphics Processing Unit.

HRRS: High-Resolution Remote Sensing.
HSI: Hyperspectral Imaging.

KNN: K-Nearest Neighbors.

LU: Landscape Units.
LULC: Land-Use and Land-Cover.

ML: Machine Learning.
MSI: Multispectral Imaging.

NBR: Normalized Burn Ratio.
NDBI: Normalized Difference Built-up Index.
NDMI: Normalized Difference Moisture Index.
NDVI: Normalized Difference Vegetation Index.
NDWIF: Normalized Difference Water Index of McFeeters.

OA: Overall Acuracy.

PCA: Principal Component Analysis.
PRISMA: Preferred Reporting Items for Systematic Reviews and Meta-Analysis.

RGB: Red, Green, Blue.
RS: Remote Sensing.

SLR: Systematic Literature Review.

SMOS: Soil Occupation Monitoring System.
SVM: Support Vector Machine.
SWIR: Short Wave Infrared.

THEIA: Theia Land Data Centre.

VNIR: Visible and Near Infrared.

CHAPTER 1

Introduction

Numerous aspects of land governance and sustainable development rely on Land-Use and Land-Cover (LULC) mapping through the classification of Remote Sensing (RS) images [**1**, **2**]. This classification allows government entities and companies to make informed decisions regarding its use.

For example, in the context of urban planning [**3**, **4**], agriculture [**5**–**8**], fire prevention [**9**, **10**], or even mineral exploration, the use of LULC mappings allows for better planning. In addition, it aids in comprehending the planet as a system; despite being frequently used interchangeably, land cover and land use are not synonymous. Land cover refers to the types of features present on the earth's surface, including, among others, trees, rocks, lakes, and roads. However, land use refers to the specific activity or function being developed on that piece of land, such as urbanization or residential development. Comparatively, when referring to a neighborhood, the term land cover would refer to the road, roofs, grass, and trees, whereas the term land use would refer to residential use. The significance and value of these High-Resolution Remote Sensing (HRRS) images have grown as their technology, availability, and quality improved.

Moreover, because of the opportunities they afford, they provide access to vast quantities of data that span vast areas of land in detail and over extensive periods [**11**]. Therefore, most land studies involve analyzing satellite or aircraft-mounted sensor photographs. Nevertheless, the size and variety of data types in these images made them challenging to process, further complicating the situation. A single pixel may hold spatial, spectral, and geometric data, among other things.

The process of acquiring and analyzing LULC data was enhanced by the combination of Machine Learning (ML) and the development of computer vision technology. Because technology enables coverage, detection of features, and planning on a scale that would be impossible with human ground coverage, a paradigm shift is occurring.

This chapter will introduce the dissertation's subject, which is the application of Convolutional Neuronal Network (CNN) for land cover classification from HRRS images. The introduction starts with the motivation for this work, followed by the aims and questions to be answered. The thesis structure will also be addressed.

1.1. Background and Motivation

With the introduction of computers in the 20th century, numerous opportunities and possibilities surged. Since the 1940s [**12**, **13**], humans have had an interest in trying to replicate their learning and decision-making capabilities on computers using what they

later referred to as Artificial Inteligence (AI), which includes concepts such as ML. This concept of ML has become more popular with time due to its ability to" learn" and make predictions from data.

The neuronal network (NN) is one of the most significant developments in this context. These models are based on the biological characteristics and functions of the brain (hence the term neuron). In the context of this thesis, the most important types of NN are Artificial Neuronal Network (ANN) and CNNs. The ANN is primarily utilized for pattern recognition and classification and is highly adaptable. There are various types of ANN, some with a simpler architecture (typically referred to as shallow) and others with a more complex architecture (deep). CNN exemplifies one of these more complex and sophisticated types of ANN. These deep learning architectures specialize in the handling of grid-like data, as is the case of images (the data here used), because they can establish and capture patterns in the raw input data without as much pre-processing, a big difference when compared to a more common ANN, which typically isn't capable of extracting features from the data, needing more preprocessing for similar tasks.

During the 1950s [**14**] and 1960s [**15**], the early development of AI and neuronal network architectures laid the foundation for what is more commonly used today. Some investigators could make progress at the time. They faced, however, a problem related to the difference in the amount of processing power they needed to continue the development of their studies and the one that was available. This led to a stalling in this field in the next decade, the 1970s, that would be solved later.

It would be only in the 1980s and 1990s that the discovery and design of new algorithms reignited interest in these technologies. As neuronal networks became more rapidly developed, they began to compete with conventional algorithms and eventually supplanted them. This change occurred thanks to the amount of data available and the increase in computational power. This was also around the time that some of the components that would later influence the CNN we know today began to emerge [**16**]. This novel neural network was primarily used for image processing [**17**, **18**] and has since become the state of the art in image analysis.

The adoption of these architectures has, however, only been widespread for the common public (outside of the scientific community) in the past decade. This was due to two primary reasons: the increase in computer power through the availability of the Graphics Processing Unit (GPU)), which made the use of deep learning (which includes CNNs) more practical and efficient. Other than that, their performance in the ImageNet Large Scale Visual Recognition Challenge started in 2010 [**19**]. In this context, CNNs have dominated since 2012 [**20**], causing even more interest in deep learning algorithms.

These algorithms are particularly relevant in the context of this thesis due to the characteristics of the data provided by the DGT, that is an HRRS with several specific characteristics. This data has multiple bands, which has the potential for redundancies and limited variance between adjacent bands.

1.2. Research Questions

Given the environment and motivation of this thesis, the emphasis will be on the development of a system capable of automatically recognizing and categorizing ground types based on Multispectral Imaging (MSI) data supplied by the Direção Geral do Território (DGT) (captured and shared by the European Space Agency (ESA) within the Sentinel-2 mission, which will be further explained in Chapter 3) and under its rules. This development is aimed at answering the following research questions:

(1) How can the accuracy of the LULC classification be improved while preserving the maximum amount of information?
(2) How does the choice of CNN architecture affect the performance of terrain classification?
(3) Can transfer learning be used to improve CNN's performance on MSI data for terrain classification? What are the most effective methods for fine-tuning pre-trained models, if applicable?
(4) How do different pre-processing strategies, like normalizing and reducing the number of dimensions, affect the performance of CNNs when using MSI data to classify terrain?

1.3. Goals

A variety of objectives will be addressed throughout the course of this dissertation. The primary objective of this thesis is to develop a CNN model that can accurately identify and differentiate various land cover and land use types in compliance with standards. These models will be trained using datasets obtained from satellite imagery provided by the DGT. In addition to this, one of the main goals will be to attain this outcome while minimizing computational expenses, thereby enhancing its accessibility.

In addition, a comparative analysis will be conducted between the obtained results and the existing methodologies employed for the classification of said terrains to conduct an in-depth investigation of the outcomes.

1.4. Methodology/Development Process

The research process employed the Design Science Research (DSR) methodology [**21**]. DSR is a concept that focuses on the advancement of design ideas, methodologies, and tools. The selected methodology was chosen based on its inherent characteristics.

The process was structured according to the steps outlined in Figure 1.1; nevertheless, there is no expectation that researchers would consistently do activities 1 through 6 sequentially.

In the initial phase of applying DSR to a specific environment, four options can serve as the entry point:" problem-centered approach"," objective-centered solution"," design and development-centered approach", and" client/context approach. The "problem-centered approach" was selected for this study due to the pre-existing definition and identification of the problem.

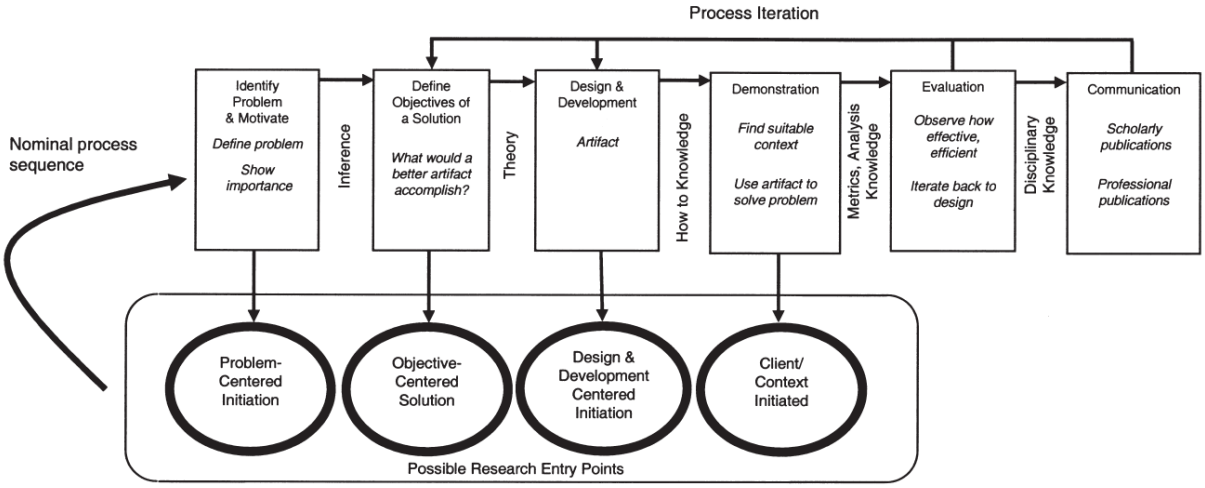


FIGURE 1.1. DSR process model
Source: [**22**]

The research in this case was conducted following the below-listed activities.

(1) Identify the problem and motivate: this is done in the introduction and the literature review.
(2) Define the objectives of a solution: are all done in Section 1.3 based on the literature read.
(3) Development of a solution to the identified problem. This will be done through the development of mechanisms capable of performing the automatic classification of territory based on MSI.
(4) Demonstration: The evaluation of the previously developed mechanisms is conducted in this stage, utilizing a subset of the data sets provided by the DGT. The datasets must include samples used for testing only, meaning that they have not been included in prior phases.
(5) Evaluation: After implementing the designed processes, results will be gathered. By analyzing these outcomes, it can determine how effectively it performs and where it might be improved. Can also be compared to other methods for the same purpose to better understand its precision and processing time.
(6) Communication: The final stage of the communication process involves the development of the thesis and the presentation of the completed work.

1.5. Dissertation Structure

The present thesis consists of six chapters with the following contents:

- 1 presents the topic of study and its motivation. In addition, it is also here that the research objectives are defined, and the methodology is first described.
- 2 is divided into two major parts. The first is a contextualization of the main terms and technologies approached in the thesis, namely, AI, ML, CNN, HRRS, LULC, and MSI. The presentation of the reviewed literature, as well as the methodology used to conduct this literature review, comprise the second section.

- 3 provides a comprehensive overview of the dataset, from how the data was collected and preprocessed to what was done with the DGT data. In addition, explains the implementation, development, and testing of CNN architectures, as well as the transfer of learning.
- 4 starts by describing the performance assessment metrics used for CNN testing. In addition, we will present the outcomes of the experiments, along with some observations and an analysis of them.
- 5 explains the interpretation of the results presented. In addition, it outlined potential future enhancements to the developed work based on the previously described findings. essentially represents the thesis's conclusion. A comprehensive summary of the completed work that attempts to answer all of the questions raised in the introduction, along with some practical implications of the developed work and potential future work.

CHAPTER 2

Literature Review

This chapter of the thesis will demonstrate an understanding of the background concepts as well as the state of the art of related technologies and strategies for problems that are comparable to those that were encountered. It will also explain how the solutions that were listed function.

2.1. Background Concepts

As stated previously, this section will provide background information on some of the technologies that will be used or discussed in this dissertation.

2.1.1. Machine Learning

ML is a subfield of AI that enables software systems to predict future events without being explicitly programmed to do so. To predict or estimate output values, ML algorithms utilize historical data as input. For these predictions to be accurate, substantial amounts of data are required. Generally speaking, ML methods can be classified according to their complexity and their specific learning strategy type, as depicted in Figure 2.1. When discussing complexity (typically when talking about neuronal networks), it is essential to distinguish between shallow and deep learning. Considering the type of learning, the three primary categories are supervised learning, unsupervised learning, and reinforced learning [**23**].

The data involved is the primary distinction between supervised and unsupervised learning types. Before training, it is necessary to preprocess the data in the context of supervised learning. Typically, the data is annotated with labels, and the model acquires knowledge of patterns during the training process to reproduce them later. As a result, a larger amount of data is required for tasks such as recognizing letters or animal species, which is essentially a classification problem (i.e., of models, NN, which can be shallow or deep learning, depending on the number of layers, Support Vector Machine (SVM) [**24**], etc.). Unsupervised learning focuses primarily on datasets that lack labels and, as a result, does not require as much manual preprocessing, necessitating that the architecture establish connections and relationships within the data autonomously. The discovery of hidden patterns or structures frequently results in the formation of clusters and groups within a dataset (i.e., K-Means clustering, Hierarchical Clustering, etc.). The other type of learning that has not yet been discussed is reinforced learning, where the architecture consists primarily of feedback-receiving learners. Each time an action is carried out, it receives feedback to determine whether it is accurate or not. The system consistently

seeks to maximize precision, which facilitates its growth and development (i.e., Q-learning, Policy Gradient, etc.).

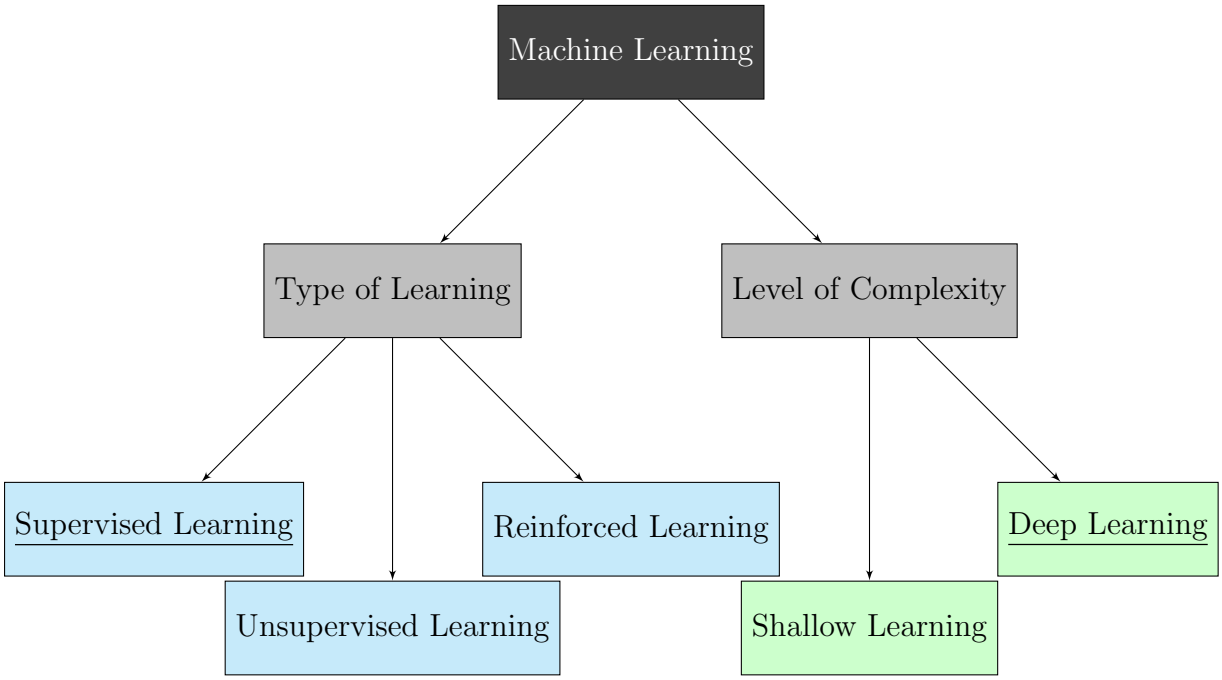


FIGURE 2.1. Classifications of Machine Learning

In what relates to the complexity level, as previously mentioned, a straightforward approach to distinguishing between shallow and deep learning is to examine the disparity in layers. In the field of deep learning, it is customary for the number of hidden layers to be relatively high, extending to hundreds in certain cases. In contrast, when speaking of shallow learning, this number is usually far inferior, usually a single hidden layer. As expected, this distinction between the two results in a variety of practical differences. For instance, as the number of layers grows, the architecture typically becomes more time-consuming and computationally demanding. Shallow learning can only accomplish so much, and so it is only used for simpler jobs, such as classification used in, for example, email systems for spam detection. When the case is about their deeper counterparts and the consequences of their complexity, they are capable of much more complex tasks, like picture classification or object detection.

The primary objective of this thesis is the analysis of RS data in the context of LULC classification, as previously stated. As such, and due to its superior performance when working with this type of data because of its sheer size, deep learning is the architecture employed. Of the various deep learning architectures, CNNs are one of the most frequently used in similar projects, as the reviewed papers suggest, so this justifies why they were selected in this case (deep learning is the one used and as such is underlined in Figure 2.1 along with supervised learning which is also used).

2.1.2. Applications of CNNs in image Analysis

Like other NNs, CNNs, were built and developed with inspiration from the biological functioning of the human or animal brain. This approach to supervised machine learning comprises multiple units (neurons) stacked in layers to mimic the brain's operation. A CNN is composed of the three main layer types(Figure 2.2), the input layer, the hidden layers (typically more than one in the case of deep learning, as CNNs can be hundreds), and the output layer, where by-products are returned. They are mainly used for image processing because of the way they are designed: they accept an image as input, treat it as a matrix, produce increasingly abstract versions of the image inside its hidden layers, and then return the class to which they belong or be able to segment them. As they operate based on matrixes, and images are essentially pixel matrixes, they are effective at operating them, as will be shown.

The hidden layers of a CNN are mainly of three different types. Convolutional, pooling, and the fully connected layer as shown in the Figure 2.2.

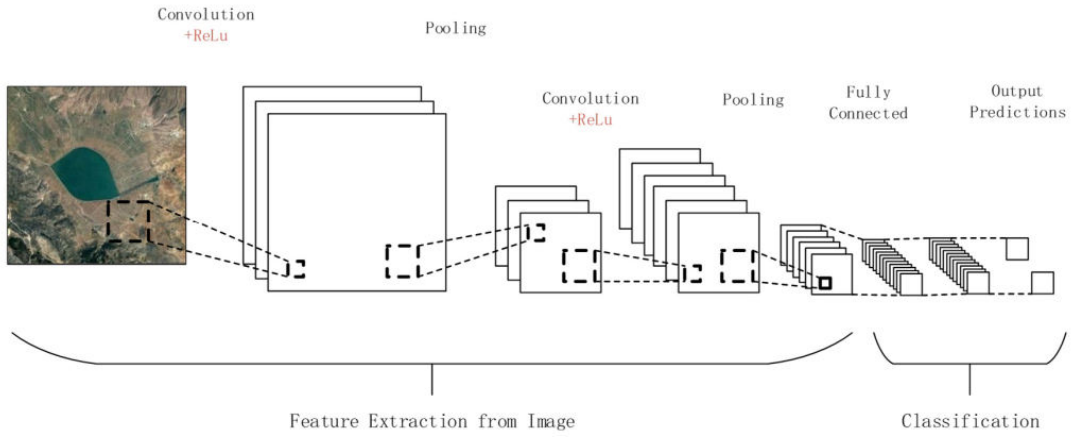


FIGURE 2.2. Example architecture of a CNN
Source: [**25**]

The convolutional layer is the fundamental component of this architecture, as it gives its name to the network itself. It's composed of three elements: the input data or input feature map, the kernel (which is essentially a weight matrix that slides over the input data), and the resulting output feature map or activation map, which is the result of the convolution between the data matrix and the kernel.

After the convolution layer, the pooling layer is typically applied. This layer is responsible for reducing the size of the acquired feature maps from the layers that came before it and are therefore a downsampling operation [**25**]. This ultimately accelerates and simplifies subsequent layers. Having a compression effect enables the extraction of the most vital characteristics while rejecting the least pertinent ones. The pooling layer can be of different types, such as max pooling, in which it retrieves only the maximum value of the area covered by the kernel (as shown in the Figure 2.4), or average pooling,

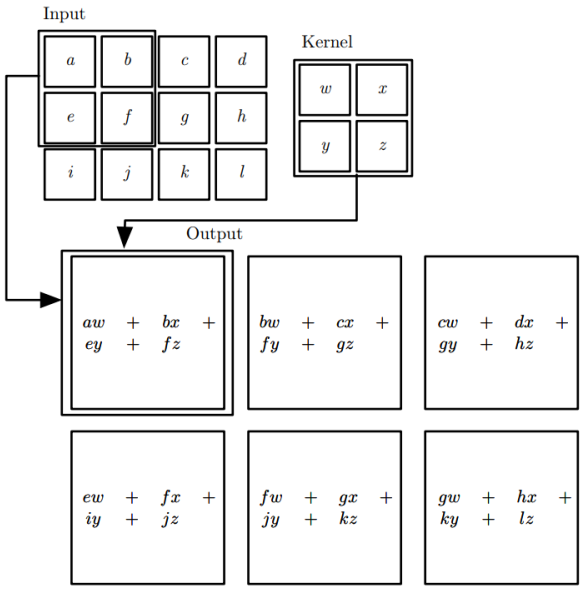


FIGURE 2.3. 2-D Convolution
Source: [**26**]

in which it retrieves the average of all the values in the area covered by the kernel. There are additional types, each with a unique application and expected outcome.

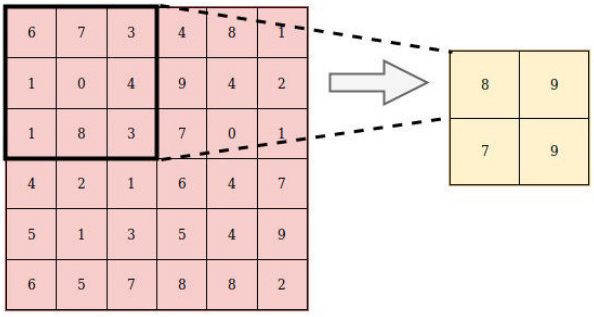


FIGURE 2.4. Max Pooling example
Source: [**27**]

It operates such that if a 6x6 matrix is subjected to max pooling with a 3x3 kernel, the result is a 2x2 matrix (Figure 2.4).

The fully connected layer connects each neuron to all neurons in the layer that precedes it and to all neurons in the layer that follows it. Typically, these layers are used to connect the final convolutional or pooling layer to the output layer in a CNN.

There are two other elements to be considered when talking about CNNs that impact their operation. The activation functions and the hyperparameters. The activation functions are functions that introduce nonlinearities to the network, making it suitable for learning more complex patterns. They essentially calculate the output of a neuron based on the input. The two most commonly used activation functions are ReLu (Rectified Linear Unit)(in the hidden layers) and SoftMax (in the output layer). Both can be seen in the Figure 2.5.

The ReLu activation function is responsible for the introduction of nonlinearity by returning its input value as long as it is positive. If not, it returns 0. With the following formula:

$$\text{ReLu: } f(x) = \begin{cases} 0 & \text{if } x < 0, \\ x & \text{if } x \geq 0. \end{cases} \tag{2.1}$$

The Softmax is usually used at the end of the neuronal network with the fully connected layer because it allows the transformation of the outputs into probabilities for each of the possible classes. The sum of all the probabilities will be 1, meaning that those probabilities are normalized. With the formula:

$$\text{SoftMax: } f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}}, \quad \text{for } i = 1, \ldots, n. \tag{2.2}$$

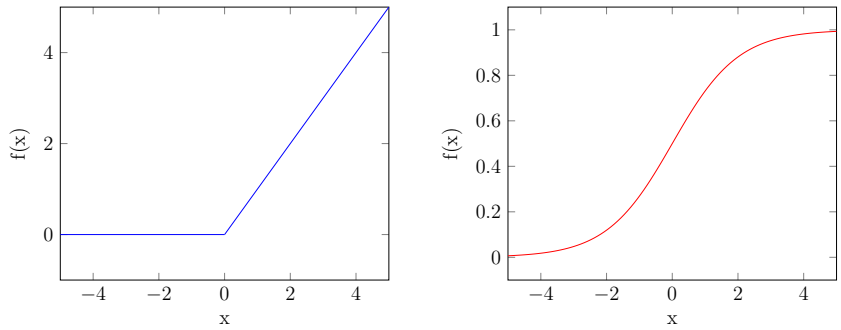


FIGURE 2.5. ReLu and SoftMax Functions

The other referred element, hyperparameters, are essentially the user-defined parameters of the network that are not automatically defined by the network. They have a direct impact on the results and computation time and are subject to fine-tuning because they will likely not be optimal on the first attempt. These include the size and number of kernels, the existence of padding in the convolution layer, the number of neurons, and the activation functions.

2.1.3. Multispectral Imaging

Multispectral imaging plays a very important role in modern remote sensing, which makes it possible to explore and get quite a bit more information about Planet Earth than when using traditional Red, Green, Blue (RGB) images. Essentially, MSI captures images of an object at different wavelengths, in this case, the earth, making it possible to study it from a spatial and spectral perspective. These different wavelengths can range from infrared to ultraviolet. Along with MSI, there is another type of image that works with the same fundamental theory, Hyperspectral Imaging (HSI). The main difference is that the number of captured wavelengths in the HSI is far superior; MSI typically has no more than fifteen, while HSI can have hundreds. This idea can be easily understood through the analysis of the Figure 2.6, which compares an image that everybody is used to, RGB

is composed of three different bands, red, green, and blue, with one of these images with a higher number of bands. These images, however, require specific optical sensors to be captured, usually a spectrometer.

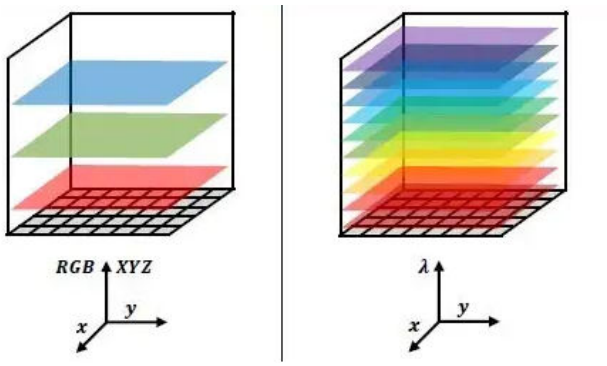


FIGURE 2.6. Comparison between a RGB and HSI images
Source: [**28**]

In the RGB case, Figure 2.6 demonstrates the distinction between the three expected bands and the numerous MSI bands on the right. Each of these bands, some of which may even coincide with RGB, collects and gives information about the presence of vegetation, water bodies, and other possible things. RGB is much more limited in comparison. This number of bands and the sheer amount of data are what justifies the use of machine learning, and in this case, deep learning. Due to its capability to handle high-dimensional data in large amounts.

2.2. Related Works/Projects

This section will provide a summary of the articles analyzed on this topic and their diverse approaches to the issue at hand, which may serve as good examples of how to address the situation.

2.2.1. Systematic literature review

In the development and writing of this thesis, the methodology in order to do the Systematic Literature Review (SLR) will be used is Preferred Reporting Items for Systematic Reviews and Meta-Analysis (PRISMA), which was proposed by Barbara Kitchenham in [**29**].

The review's article selection method is based on the three phases depicted in the Figure 2.7: identification, screening, and inclusion. B-On, a virtual library that provides access to a massive number of scientific publications, journals, and ebooks (from editors like IEEE, Springer, etc.), was utilized for the majority of the research. Some of the papers read at this point came from the references in some of the articles that were researched.

In order to perform research from this library, the following keywords were employed: CNN, MSI, terrain classification, neuronal network, deep learning, artificial intelligence, multispectral, Sentinel-2, and satellite images.

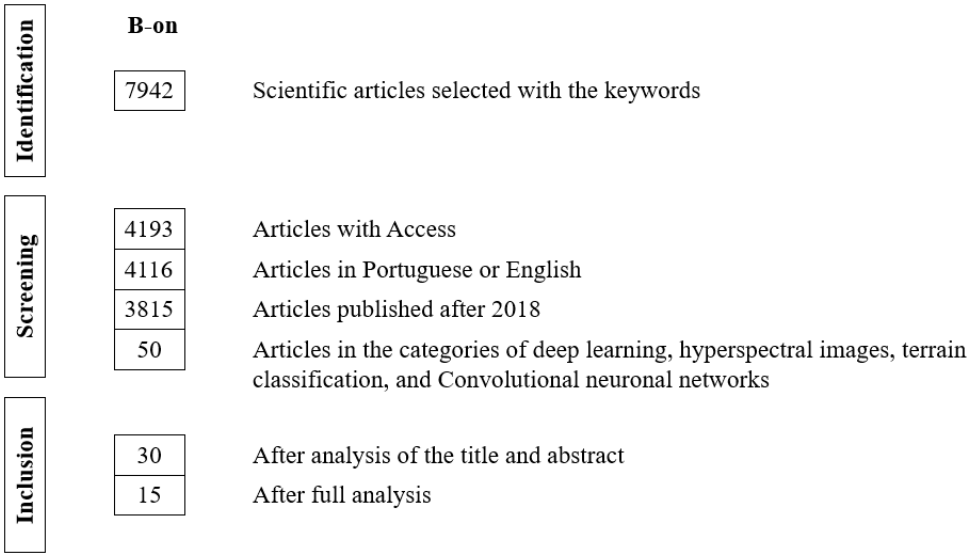


FIGURE 2.7. Prisma filtering process
Adapted from PRISMA (Page et al.2021)

To further narrow down the number of publications, a list of inclusion and exclusion criteria was established (as shown in Table 2.1).

TABLE 2.1. Exclusion and Inclusion Criteria for Similar-Work Searches

Exclusion Criteria
• Documents that aren't articles
• Written in a language other than Portuguese or English
• Duplicate articles
• Articles from before 2018
Inclusion Criteria
• Articles that discuss the topic of hyperspectral pictures
• CNNs and neural networks applied to MSI and HSI-related articles
• Articles on the subject of terrain classification using CNN and HSI
• Free or inside ISCTE's scientific license articles

After applying the exclusion criteria, a total of articles 50 were left. The inclusion criteria were then applied after reading the title and abstract of each article, resulting in a total of papers being selected for further study. The final phase involved a thorough examination of these articles, of which 15 were selected for inclusion in the systematic literature review of this thesis. It's worth noting that this number has since this stage increased during the course of dissertation work.

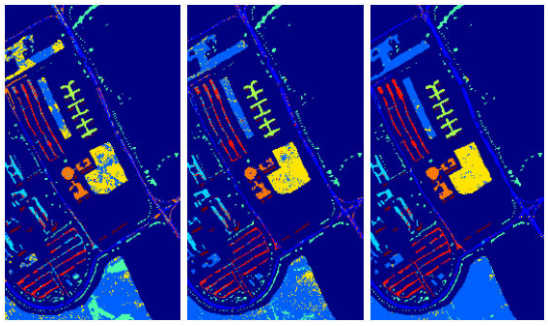
2.2.2. Previous Studies in Land Cover Classification

In this dissertation, a CNN capable of identifying and classifying MSI terrain types will be developed. This algorithm may be useful to the DGT because it may be superior to what they have already implemented. Although MSI data will be used for the search, HSI-using articles were considered to expand the available options.

When considering HSI, the documents described in this section were considered as the most relevant. Using three distinct algorithms, [**30**] classifies terrain from the HSI (Pavia University Center dataset Figure 2.8a) using two shallow learning-supervised algorithms, K-Nearest Neighbors (KNN) and SVM, and one deep learning algorithm, 3D CNN with the results shown in Figure 2.8b. This 3D-CNN [**31**] signifies that the kernel, rather than having to be applied to every pixel of a given layer and running on all layers, will be applied to all layers for each pixel simultaneously, resulting in 3D convolution that takes spectral and spatial features into account, as demonstrated in the figure. Due to its operation, this CNN type may be the most suitable choice for this type of image processing. The Figure 2.8 displays the results that were obtained.

#	Class	Samples
1	Asphalt	6631
2	Meadows	18640
3	Gravel	2099
4	Trees	3064
5	Painted metal sheets	1345
6	Bare Soil	5029
7	Bitumen	1330
8	Self-Blocking Bricks	3682
9	Shadows	947

(A) Data and Classes



(B) Segmented w/ KNN, SVM, and 3D-CNN, from L-R

FIGURE 2.8. Classification and Segmentation example test in Pavia University dataset

Source: [**30**]

As expected, the 3D-CNN had the highest accuracy, followed by the SVM, and the KNN had the lowest accuracy for the reasons stated previously. Comparatively, the execution times of the various KNNs, SVMs, and 3D-CNNs varied. The authors conclude in their paper that implementing preprocessing techniques, such as Principal Component Analysis (PCA), can increase CNN's accuracy and decrease its training time.

During the creation of the CNN, three distinct activation functions were evaluated: Tanh, Sigmoid, and ReLu. Among these functions, Tanh demonstrated the best performance in terms of both speed and accuracy. In this instance, the CNN was trained and created from scratch; however, in many cases, "transfer learning" is employed (as in [**31**–**33**]), which entails the use of pre-trained CNNs with only structural modifications, typically centered on the fully linked layers.

In a different publication [**31**], the authors propose employing multiple CNN types, including one that combines 2D and 3D CNNs, to achieve superior results, and are called HybridSN (Figure 2.9). The structure of this CNN and has been introduced previously in [**34**, **35**].

As mentioned earlier, the integration of PCA plays a crucial role in reducing redundancy within HSI data before it is fed into the models. The impact of varying PCA

FIGURE 2.9. HybridSN model
Source: [**31**]

ratios on Overall Accuracy (OA) is explored in [**35**]. Additionally, data augmentation techniques are employed to increase the number of training data points, further reducing the risk of overfitting post-PCA.

The data processing pipeline encompasses two distinct convolutional stages: the 3D stage, which will extract spectral-spatial features from the various spectral bands, and the 2D stage, which will identify the abstract spatial characteristics.

In addition to this neural network, we deployed a 3D ResNet [**36**], a 2D CNN, and a 3D CNN. HybridSN possesses the highest OA, followed by 3D-ResNet, 3DCNN, and 2D-CNN. All other architectures fall within the 95th percentile, but the 2D-CNN architecture falls within the 91st percentile, making it the least desirable option.

Additionally, as demonstrated in [**34**], HybridSN has significantly faster training and inference times compared to 3D-CNN alternatives, offering a promising results with the combination of high accuracy and computational efficiency. This is particularly advantageous when dealing with images with high spectral dimensions, as is the case with Hyper-Spectral Imagery (HSI).

A comparison of the accuracy and efficiency of CNN and ANN (often multilayer perceptrons, or MLP) in the context of HSI has already been conducted. As predicted, however, these shallow learning architectures have encountered difficulties when dealing with such images. The absence of convolutional and pooling layers increases the likelihood of overfitting and performance issues.

Despite previous literature suggesting the potential benefits of combining 2D and 3D CNNs, our dataset's characteristics aren´t quite the same. Our focus is on MSI, rather than HSI, so the primary goal will be the development with this in mind, considering then architectures which are ultimately 2D in nature.

In the context of handling MSI imagery and probably in the whole of this thesis, the most important article is the one that dealt with the dataset mentioned in [**37**], the BigEarthNet. This is a dataset created at the Berlin Technical University, which consists of 590,326 pairs of Sentinel-1 and Sentinel-2 patches from images. The Sentinel-2 is the most relevant, as is data from the same satellite as the one provided by the DGT. These 590,326 image patches are from the same approximate period as the ones the DGT provided between 2017 and 2018. Using these patches of labeled (according to the 44 CLC2018 classes) Sentinel-2 image, tests were conducted to check the performance of the use of CNNs to handle and classify these images. This articles primary conclusions focus on the positive results achieved with the use of CNNs that were designed and trained from

scratch when in comparison with transfer learning based models that use the ImageNet weights underperforming in comparison.

Another article, [**32**], focuses on the classification of Sentinel-2 images, this time on a different dataset, the EuroSAT, using 10 different classes. This different in the number of classes help improve the performance of the model probably and it achieved the 88.21% average accuracy.

In [**38**, **39**] the focus of the tests isn't as much in model, it is in the band selection within the imagery for classification. As the Sentinel-2 images originally have 13 bands, some of these are more or less important, both dependant on their respective spatial resolution as from the different wavelengths they capture, these papers focus on doing tests with different band combinations and different models. Conclusions indicate that datasets usually achieve better results, with [**38**] achieving accuracy of 94.8% with only 4 bands and an accuracy of 88.8% with 10 bands. [**39**], despite considering only 5 classes 5 classes, achieves a accuracy of 98.1% when using a combination of the three RGB bands with NIR.

Moving to [**40**], this article focuses on land cover mapping in Algeria, particularly in the context of natural disaster prevention. A CNN with a pretty shallow architecture, featuring only 2 hidden layers, is employed to analyze Sentinel-2 imagery of the region. Object-based analysis is also incorporated into the classification process. The results are very positive, with an achieved accuracy of 93%, marking a clear improvement over alternative methodologies that featured Support Vector Machine (SVM) classifiers.

Throughout the literature review, various image-processing techniques utilizing CNNs were discussed. HybridSN, frequently mentioned in the context of HSI due to its efficacy in handling multi-layer data, it is however not necessary for our MSI dataset. Our approach will be closer to the models after shown, like the treatment of the BigEarthNetAs. As result, the choice was the 2D-CNN. Nevertheless, transfer learning will also be evaluated, although some articles suggest it may not outperform custom-trained models from scratch.

The text also mentions that a comparison between CNN and an ANN has been conducted and that the latter has difficulties because it lacks convolutional and pooling layers.

CHAPTER 3

Dataset Overview and CNN Development

This chapter will include a comprehensive review of the work executed and the methodology employed in its development. To categorize land based on the defined classes established by the DGT, a series of experiments were conducted using a dataset provided by this organization. This dataset encompassed MSI data for a specific location inside Portugal. The main goal was to develop a neuronal network that could present something different and have better overall classification results in comparison to what is currently being used (shallow learning).

Taking this into consideration, and in a concise overview of the undertaken actions, the initial measures involved attaining a comprehensive comprehension of the accessible data. Then, the data was treated in such a way that it could be divided and fed into the CNNs for training, testing, and validation to achieve the best possible solution that would facilitate the best overall accuracy, the lowest loss, and without forgetting to try and optimize the time it took for it to work. Once the data had been prepared, the subsequent stage involved the preliminary establishment of the CNN. To do this, basic architecture models were considered (due to the specifications of the data, the neuronal network couldn't have many layers), and an initial framework was established. From there, diverse experiments were conducted to enhance the performance of the original CNN. There were also trials conducted involving transfer learning, which refers to the utilization of the deep portion of pre-trained CNNs while modifying only the topmost classification layers. These experiments were done to compare and check what approach would achieve the best results in classifying the given data.

The chapter will be divided into several subchapters, outlining all the steps and tests done to and with the data to achieve the results that will also be shown.

3.1. Data Collection and PreProcessing

As previously stated, the data used in this thesis was provided by the DGT. The provided data is relative to an area of the Portuguese mainland (further explanation provided in 3.2) and will now be explained in the following 3.1.1 and 3.1.2 how the data was collected and pre-processed by this entity.

3.1.1. Sentinel-2 Data

The European Commission, more specifically the ESA, owns the Sentinel-2 constellation of satellites used in the mission of the same name to collect data. This mission was launched alongside others with the same name (Sentinel), distinguishable by their number (in this case, the 2). The Copernicus program essentially has the function of being the earth

observation component of the EU space program. With the help of satellite constellations and other sources, the aim of this program is to collect reliable and accurate remote sensing data that can be used, for instance, to monitor environmental changes and global warming.

When addressing about the Sentinel-2 mission, it is important to consider the resolution (or granularity) of data acquisition along four dimensions:

- Temporal Resolution;
- Spatial Resolution;
- Spectral Resolution;
- Radiometric Resolution;

The temporal resolution is related to the time it takes for a satellite to revisit the same area. In the case of the Sentinel-2 constellation, as it is composed of the two mentioned satellites positioned with a difference of 180^{o}, the temporal resolution is 5 days, each having a 10-day cycle. The spatial resolution is defined by the MSI instrument and depends on the specific sensor configuration. This resolution is considered when talking about the at-ground representation of the Earth's surface by a single pixel in the satellite images. The spectral resolution is the instrument's capability to differentiate features along the electromagnetic spectrum, which makes it possible to detect much information about the Earth's surface. The final resolution considered is the radiometric resolution. The radiometric resolution of the instrument is defined as the capacity a system has to detect variations in intensity or reflectance. For the MSI instrument, the radiometric resolution is 12 bits, making it possible to represent intensity values up to 4095. These intensities are, however, converted to reflectance and represented as 16-bit integers in the final products.

The Sentinel-2 mission, composed of Sentinel-2A (launched in 2015) and Sentinel-2B (launched in 2017), has the purpose of collecting multispectral optical data (see Table 3.1) using the two already in-service satellites, Sentinel-2A, launched in 2015, and Sentinel-2B, launched in 2017. The mission will soon be expanding its constellation, with two further launches planned for the next two years, 2024 and 2025, of Sentinel-2C and Sentinel-2D. The data collected by this mission has various uses, for example, urban planning, crop monitoring, or forest monitoring.

Each of these satellites is outfitted with an MSI instrument containing thirteen bands, the characteristics of which are detailed in the Table 3.1.

As observable in the Table 3.1, the spatial range has three possible values: 10m, 20 m, and 60 m. The 10m spatial resolution has the particularity of being used in collaboration with other missions (Landsat-8 (an American space program with similar goals) and SPOT-5 (operated by the French Space Agency and with comparable objectives)) with a focus on land classification. Both satellites were fully developed and designed by Airbus Defense and Space in France. The capture wavelength range goes from the visible to

TABLE 3.1. Original Sentinel-2 Bands and Resolutions

Band Number	Spatial Resolution (m)	Central λ (nm)	Spectral Width $\Delta\lambda$ (nm)	Description
1	60	443	20	Ultra Blue
2	10	490	65	Blue
3	10	560	35	Green
4	10	665	30	Red
5	20	705	15	Visible and Near Infrared (VNIR)
6	20	740	15	Visible and Near Infrared (VNIR)
7	20	783	20	Visible and Near Infrared (VNIR)
8	10	842	105	Visible and Near Infrared (VNIR)
8a	20	865	20	Visible and Near Infrared (VNIR)
9	60	945	20	Short Wave Infrared (SWIR)
10	60	1375	30	Short Wave Infrared (SWIR)
11	20	1610	90	Short Wave Infrared (SWIR)
12	20	2190	180	Short Wave Infrared (SWIR)

the Visible and Near Infrared (VNIR) and Short Wave Infrared (SWIR) (as shown in Figure 3.1 and accordingly also in Table 3.1).

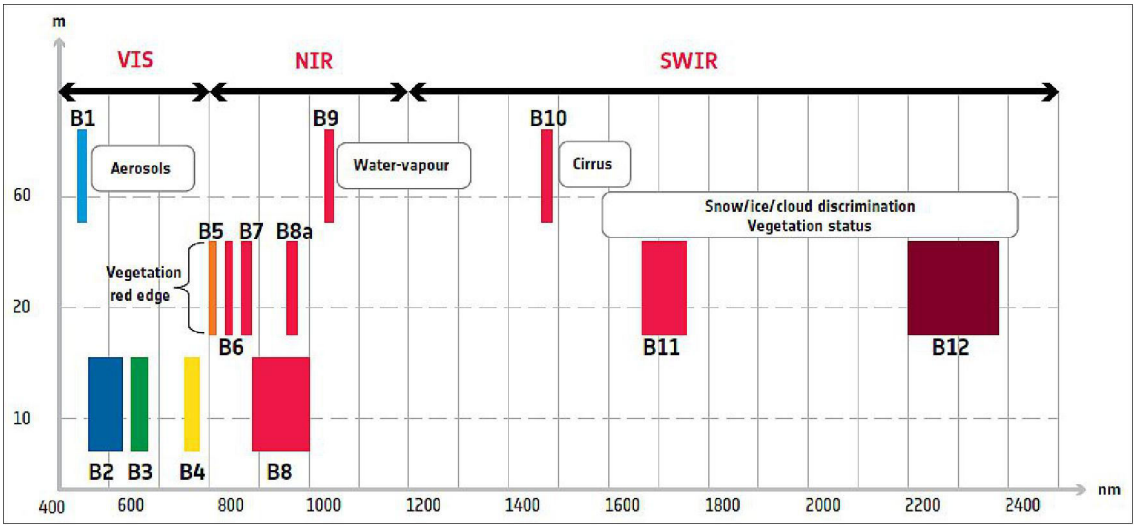


FIGURE 3.1. Sentinel-2 Bands (Image credit ESA)

Each different combination of spatial and spectral resolution has its own goal. The 10m bands focus mostly on the visible light spectrum due to its sensitivity to the chlorophyll in vegetation, both the absorbed and the total in the vegetation itself as well as the soil background. The 20m bands are mostly in the VNIR vegetation red edge spectral domain; their main goal is applications such as snow/ice and cloud detection, along with vegetation moisture assessment and leaf area index. The other three bands of 60m spatial resolution are mostly focused on atmospheric phenomena like cloud detection and atmospheric corrections.

In addition to the MSI instrument's technical specifications, the constellation has other specifications that may impact the data. For example, the fact that satellites are sun-synchronous means that they maintain the incidence of sunlight on the Earth's surface in the image capture.

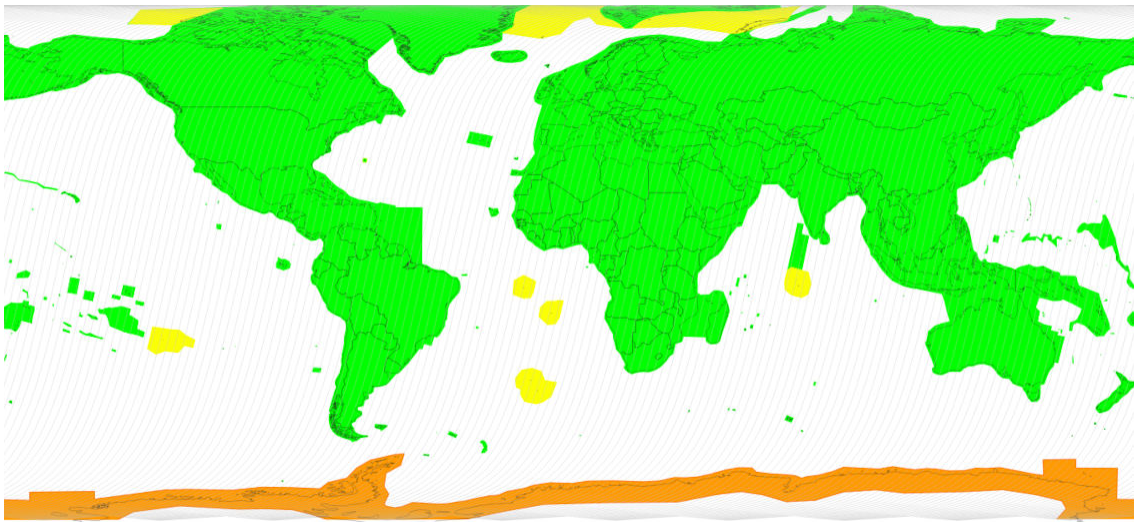


FIGURE 3.2. Sentinel-2 Mission Cover
Source: [**41**]

In the image, there are three distinct hues, each of which corresponds to a distinct period between satellite passages.

- Green - every five days
- Yellow - every ten days
- Orange - every ten days from alternate tracks.

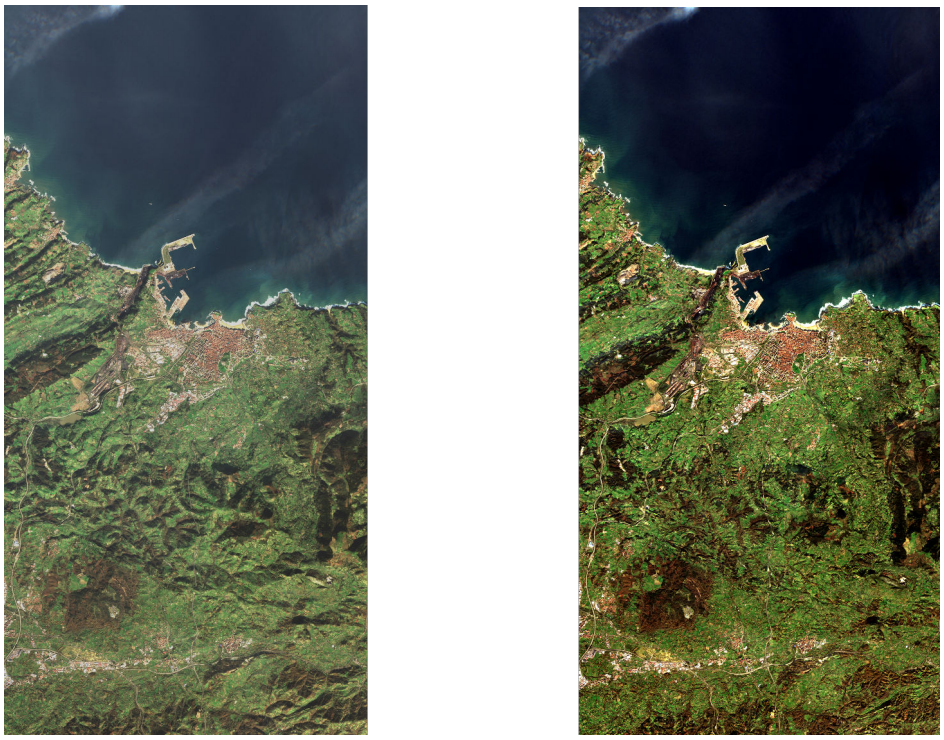
As seen in Figure 3.2, their 5-day revisiting cycle encompasses the majority of the planet.

The areas for which the products are generated can also be viewed: all continental land surfaces, inland waters between 56° South and 82.8° North, coastal islands up to 20km from the coast, islands over 100km, all EU islands regardless of size, the entire Mediterranean Sea, and all closed seas.

The product types that are accessible to users from the image collection are an additional important factor to consider for the mission. There are numerous types of products, the majority of which are not accessible to all users. Level 2A is the product type that corresponds to the provided information and is relevant. This is a type of product whose surface reflectance has been corrected for atmospheric conditions using cartographic geometry (in the Figure 3.3 is shown the difference between the level 1C and level 2A).

There is a clear distinction between the two products. Despite being preprocessed images, Level 2A (Figure 3.3b) is of higher quality than Level 1C (Figure 3.3a), to which only corrections for the upper atmosphere were applied.

The sentinel 2 data is acquired in a continuous mode known as “datatake”. The maximum length these images may take is 15000 km. All these large data sets are composed of tiles (depending on ground geometry) or granules (depending on sensor geometry). These products are the fundamental elements, with a fixed size of these data points collected over a single orbit and therefore indivisible. In the case of the products that matter to this thesis (Level-2A), their size is a square with 110x110 km2 orthoimages (images that



(A) Level 1C (B) Level 2A

FIGURE 3.3. Sentinel-2 Products
Source: [**42**]

have had their distortions corrected, distortions related to the camera and the terrain, thus providing a correct perspective of the ground shape) in projection UTM/WGS84 (the UTM relates to the projection of the map (this way considering the earth flat opposed to a globe), it divides the world in 60 different zones while keeping each zone with a constant scale and origin in the equator; the WGS84 relates to the set of parameters that is used as a reference point for spatial measurements, this one being the most used; both are needed to accurately pinpoint points on earth). They are collected in a 100-kilometer step, so there is always an area that overlaps.

However, the data used to generate the dataset and therefore provided by the DGT was not obtained directly from the ESA. It was obtained from the Theia Land Data Centre (THEIA) in France. This was because the available images have already been preprocessed with the MAJA algorithm [**43**]. This pre-processing already solves some issues; it adds a level of atmospheric correction to the bottom of the atmosphere, masks clouds, their shadows, snow, and water, and provides slope correction. This slope affects the reflectance correction to ensure that the reflectances are as they would appear on a flat surface and not a lens.

The downloaded files by the DGT were carefully filtered to include only those with a maximum cloud cover of 50%. Additionally, the bands with a spatial resolution of 60m were removed. These specific bands were primarily intended for measuring atmospheric

properties, which had already been accurately corrected for our purposes. As a result, we were left with ten bands instead of the original thirteen.

To enhance the dataset's consistency, a spatial resolution adjustment was performed on the bands captured at 20m. This adjustment involved upscaling the pixel resolution from 20 meters to 10 meters using a nearest-neighbor interpolation method. This upscaling ensured that all bands in the dataset were uniform with a spatial resolution of 10m, facilitating uniform data processing and analysis.

This new data structure with the mentioned changes is shown in Table 3.2.

Table 3.2. Sentinel-2 Bands and Resolutions post Pre-Processing

Band Number	Spatial Resolution (m)	Central λ (nm)	Spectral Width $\Delta\lambda$ (nm)	Description
2	10	490	65	Blue
3	10	560	35	Green
4	10	665	30	Red
5	10	705	15	Visible and Near Infrared (VNIR)
6	10	740	15	Visible and Near Infrared (VNIR)
7	10	783	20	Visible and Near Infrared (VNIR)
8	10	842	105	Visible and Near Infrared (VNIR)
8a	10	865	20	Visible and Near Infrared (VNIR)
11	10	1610	90	Short Wave Infrared (SWIR)
12	10	2190	180	Short Wave Infrared (SWIR)

The tiles were then used by the DGT to produce an image that would relate to a specific period that they defined. For each tile, they were able to produce the median composite of the time they intended. All the images of the period were stacked band by band, and for each of the pixels, the median would be calculated. This way, you are not affected by the effects of clouds or other atmospheric phenomena. Whenever this wasn't enough to make the data good throughout the whole image, they would then complete those period values with the use of linear interpolation with time spans that preceded and succeeded that [**8**].

Additionally, DGT derived several spectral indexes computed from the THEIA data. These spectral indexes include the Normalized Difference Vegetation Index (NDVI), the Normalized Burn Ratio (NBR), the Normalized Difference Water Index of McFeeters (NDWIF), the Normalized Difference Built-up Index (NDBI), and the Normalized Difference Moisture Index (NDMI). The formula for each is:

Table 3.3. Implemented spectral indexes with formulas

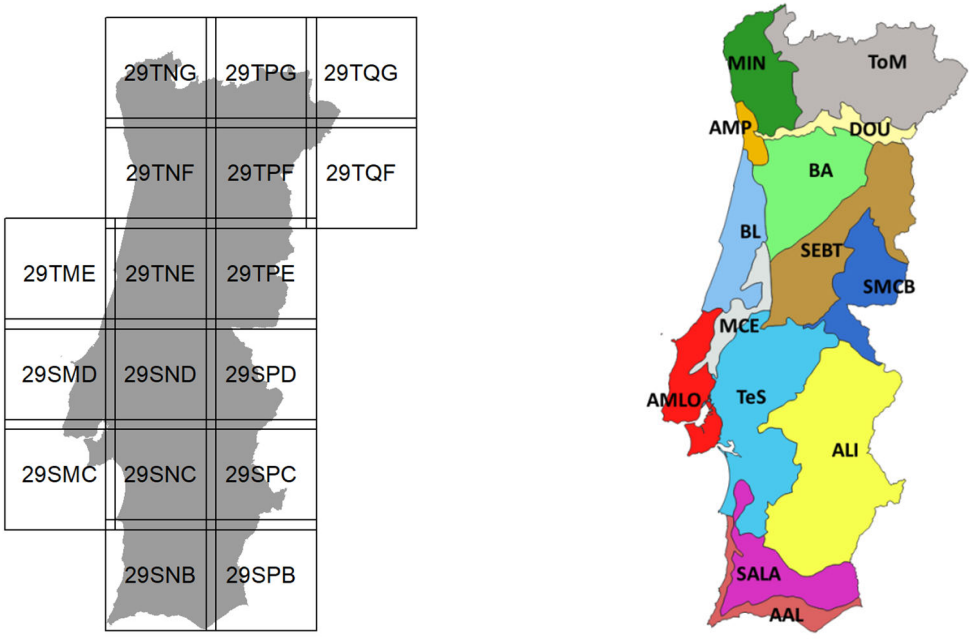
Index	Equation	Bands	Reference
NDVI	(NIR-R) / (NIR+R)	(B08 - B04) / (B08 + B04)	[**44**]
NBR	(NIR-MIR2) / (NIR+MIR2)	(B8A - B12) / (B8A + B12)	[**45**]
NDWI	(G-NIR) / (G+NIR)	(B03 - B08) / (B03 + B08)	[**46**]
NDBI	(MIR1-NIR) / (MIR1+NIR)	(B11 - B8A) / (B11 + B8A)	[**47**]
NDMIR	(MIR1-MIR2) / (MIR1+MIR2)	(B11 - B12) / (B11 + B12)	[**48**]

For both GeoTiffs and spectral indexes, data quantiles were data were calculated for each of the 10 band time spans. The quantiles 10, 25, 50, 75, and 90, as well as the q75-q25 and q90-q10 quantile differences.

3.1.2. Auxiliary Data

In addition to the GeoTiff files containing the collected derived products (Sentinel-2 data), the dataset also included DGT files containing training points. These points correspond to the labels for certain locations, which correspond to established classes that will be used to classify the land according to the convention further down the line. There are two formats for these points: geopackage (gpkg) and comma-separated values (CSV). Both files contain essentially the same information, the difference being the way the coordinates for each of the points are stored. With the UMT projection and WGS84 parameters, the coordinates in the gpkg are generated in a manner comparable to that of the Sentinel-2 imagery products. The CSV file contains points with planimetric coordinates (X and Y). These coordinates are comparable because the UMT projection is a type of planimetric coordinate system (planimetric coordinates are those that consider the map to be flat and stabilize the positions; accordingly, the UMT projection is a type of planimetric coordinates).

Given that the DGT is providing the data in this instance, the focus is on mainland Portugal. From the perspective of Sentinel-2, the country land coverage can be divided into the seventeen tiles shown in Figure 3.4a. From a administrative standpoint, the division is divided into fourteen distinct Landscape Units (LU), as shown in Figure 3.4b.



(A) Tiles (B) LU

FIGURE 3.4. Portugal map from a tile and LU perspective
Source: Adapted from the provided documentation [**49**]

These landscape units were defined during the production of the COSsim (Land Occupation Letter) production process, specifically for COSsim 2018 [**49**]. COSsim is an experimental product developed by DGT to partition mainland Portugal into small zones. These zones are identified by unique letter codes detailed in Table 3.4. These letter codes are shown in Figure 3.4b.

TABLE 3.4. Characterization of the LU used in the COSsim

*Source: Adapted from the provided documentation [**49**]*

Numeric Code	Letter Code	Name	Area (ha)	% Total area
111	MIN	Minho	49443	6
112	DOU	Douro	192796	2
113	BA	Beira Alta	792212	9
114	AMP	Área Metropolitana do Porto	88583	1
121	ToM	Trás-os-Montes	1177802	13
122	SEBT	Serra da Estrela e Beira Transmontana	928857	10
211	AMLO	Área Metropolitana de Lisboa e Oeste	389795	4
212	BL	Beira Litoral	449384	5
213	MCE	Maciços Calcários da Estremadura	217884	2
214	TeS	Tejo e Sado	1223890	14
215	AAL	Algarve e Alentejo Litoral	234797	3
221	SMCB	Serra de São Mamede e Castelo Branco	489464	6
222	ALI	Alentejo Interior	1680967	19
223	SALA	Serras do Algarve e do Litoral Alentajano	535517	6
Total			8896591	100

Along with the letter and numeric code, the Table 3.4 also shows all the names of the LU, along with the area of each zone and the percentage of the area of mainland Portugal that the zone corresponds to.

The training points, or labels, were created by the DGT for each of the LUs and not for each tile of the Sentinel-2 products. These points were created randomly for each class and each area and automatically processed to include and exclude areas of interest based on the characteristics of the land use in that area. The various classes can be seen in the Table 3.5.

The classes to the left are the training classes; a code is assigned to each of these specific classes. These classes are distinct from the final COSsim classes, which are divided into three distinct levels as shown in the table's right column. These COSsims are typically comprised of multiple training classes per level because they are less specific than training classes.

As a summary, Figure 3.5 shows the components of the dataset provided by DGT.

The dataset described in Figure 3.5 is composed of two types: the Sentinel-2 products, composed of the Sentinel-2 10-band GeoTiff, the indexes, and the metrics. The other type is the auxiliary data, which are the training points developed by the DGT according to their goals of building the SMOS.

TABLE 3.5. Training Point Classes according to the COSsim Nomenclature

		COSsim Nomenclature		
Training Points class	Code	Level 3	Level 2	Level 1
Artificialized territory	11110	100 - Artificialized	10 - Artificialized	1 - Artificialized
Industry	11101			
Road network	11120			
Oat	21110	211 - Annual Autumn/ Winter Cultures	21 - Agriculture	2 - Agriculture
Wheat	21120			
Barley	21130			
Ryegrass	21140			
Triticale	21150			
Rye	21160			
Corn	21210	212 - Annual Spring/ Summer Cultures		
Rice	21220			
Tomato	21230			
Sunflower	21240			
Irrigated Area	21201			
Vine	22101	213 - Other Agricultural Areas		
Orchards	22201			
Olive	22301			
Pasture	31110			
Cork Oak	51101	311 - Cork and Holm Oak	31 - Hardwoods	3 - Forest
Holm Oak	51201			
Adult eucalyptus	51310	312 - Eucalyptus		
Eucalyptus AA2017	51301			
1 Year Cuts Eucalyptus	51302			
Young Cuts Eucalyptus	51303			
Other Hardwoods	51410	313 - Other Hardwoods		
	51401			
Maritime Pine	52110	321 - Maritime Pine	32 - Softwoods	
	52101			
Stone Pine	52210	322 - Stone Pine		
	52201			
Other Softwoods	21310	323 - Other Softwoods		
Agricultural Spont. Herb. Veg.	31201	420 - Spontaneous Herbaceous Vegetation	42 - Spontaneous Herbaceous Vegetation	4 - Bushes and Spontaneous Herbaceous Vegetation
Mountain Spont. Herb. Veg.	31202			
Herbaceous Vegetation AA2017	31203			
Herbaceous Vegetation Cuts	31204			
Herbaceous Vegetation AA2016	31205			
Dense Bushes	61101	410 - Bushes	41 - Bushes	
Snow Bushes	61103			
Bushes AA2017	61104			
	61105			
Bushes AA2016	61106			
Bare Soil	71110	500 - Surfaces without Vegetation	50 - Surfaces without Vegetation	5 - Surfaces without Vegetation
Bare Rock	71201			
Humid Zones	81110	610 - Humid Zones	61 - Humid Zones	6 - Water and Humid Areas
Water	91110	620 - Water		

3.2. Description of the Received Data

As previously stated in the Section 3.1, the data that was provided by the DGT wasn't of the full Portuguese mainland; instead, it was relative to a specific area of the map. This particular region can be observed in the Figure 3.6, both from a tile (Figure 3.6a) and LU perspective (Figure 3.6b).

There are six tiles, designated as 29TNG, 29TPG, 29TQG, 29TNF, 29TPF, and 29QF. This region contains MIN, TOM, AMP, and DOU from a LU perspective. These, along

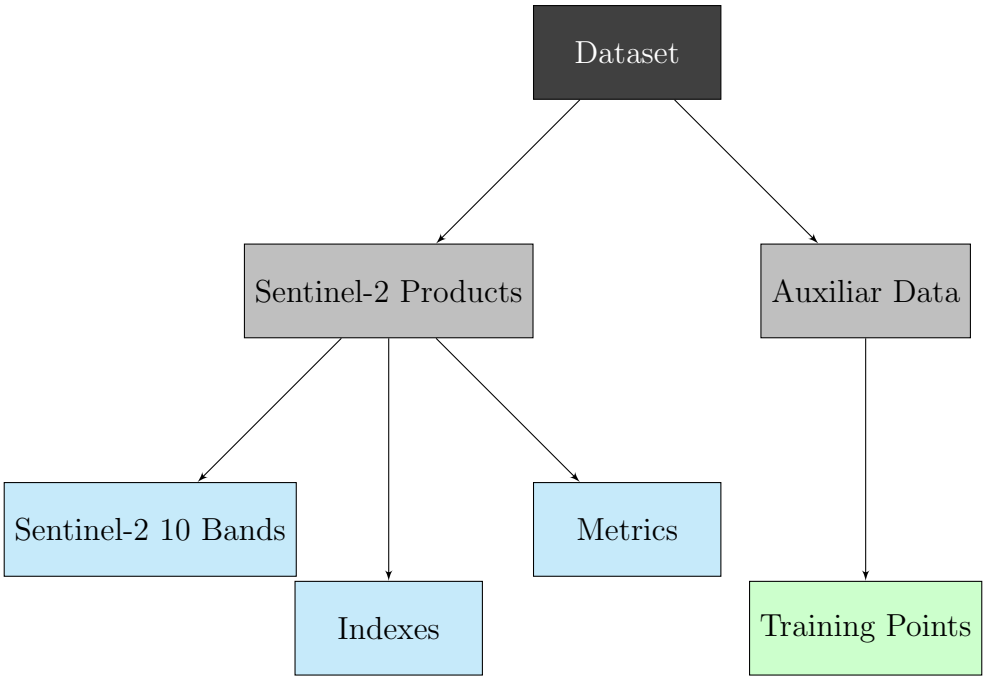


FIGURE 3.5. Summary of the provided Data

(A) Dataset Tiles (B) Dataset LU

FIGURE 3.6. Map of the provided data from a tile and LU perspective
Source: Adapted from the provided documentation [**49**]

with some specifications of their characteristics, can be seen in the Table 3.6, with characteristics such as the area of each of these LUs and the percentage they represent of the whole Portuguese mainland.

TABLE 3.6. Characterization of the LU present in the dataset

*Source: Adapted from the provided documentation [**49**]*

Numeric Code	Letter Code	Name	Area (ha)	% Total area
111	MIN	Minho	49,443	6
112	DOU	Douro	192,796	2
114	AMP	Área Metropolitana do Porto	88,583	1
121	ToM	Trás-os-Montes	1,177,802	13
		Total	1,580,624	22

In addition to the dataset details, its important to establish the relation between the tiles, which correspond to the Sentinel-2 imagery, and the LUs, which correspond to the training points. In Table 3.7, its presented the distribution of the training points across the

Level-3 COSsim nomenclature classes. The class '100:Artificialized' which is composed by the training point classes 'Artificialized Territory', 'Industry', and 'Road Network', with the best represented with 22,044 points. On the other hand the least represented class is '610:Humid Zones' with 0 samples.

Furthermore, it's worth appointing that the dataset is composed of 158,847 samples. This, when compared to the sheer size of each GeoTiff tile which are squares 10980 pixels height and width. This means that the percentage in the universe of pixels composed of the six tiles, the percentage of labeled pixels is 0.002194%.

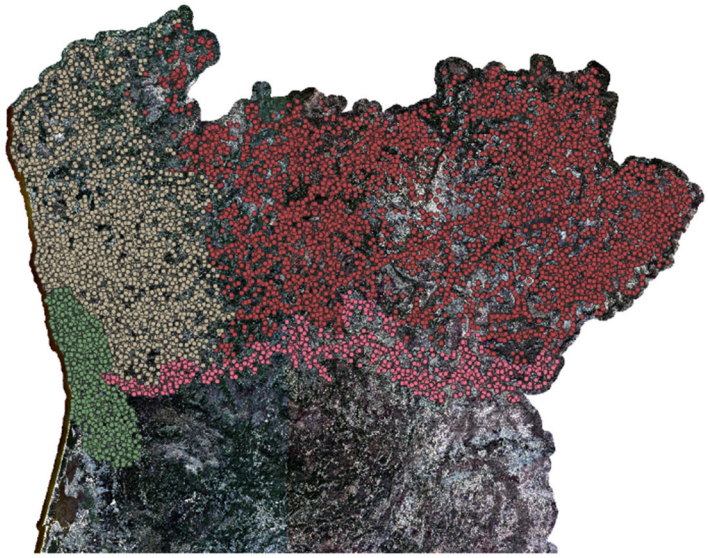


FIGURE 3.7. Overlay of the four LU training points over the six tiles

Figure 3.7 illustrates the distribution of training points across each of the six tiles. Opening a GeoTiff for each tile and superimposing the gpkg file with the training points created this image. These are the training points previously mentioned. Each distinct point color represents a unique LU, hence the four distinct colors. Several inferences can be made from the image. As expected, as the number of tiles and LUs increases, the majority of tiles consist of training points from multiple LUs. Aside from that, and only visible now, these regions only contain labels from the northernmost points. To be more complete, training points from other LU like BA (Beira Alta), BL (Beira Litoral), and SEBT (Serra da Estrela e Beira Transmontana), as shown in the Table 3.4, would be needed. It is believed that the DGT left these areas out of the dataset because the tiles wouldn't completely cover them and, as a result, they weren't useful.

3.3. Data Processing

The procedure for handling the supplied data will now be detailed. In this regard, it is crucial to recall the previously provided overview of the dataset, including the distinctions between tiles (Figure 3.4a), LU (Figure 3.4b), and training points (table something). Consequently, the applied methodology can be summed up as follows:

TABLE 3.7. Training Classes and Level 3 classes per LU from data

COSsim Level 3 Class	Training Points Class	Code	LU 111	LU 112	LU 114	LU 121	Total/ Class	Total/ level
100 - Artificialized	Artificialized territory	11110	1798	656	1000	3943	**7397**	**22044**
	Industry	11101	1798	977	1000	2793	**6568**	
	Road network	11120	1798	983	997	4301	**8079**	
211 - Annual Autumn/ Winter Cultures	Oat	21110	780	1000	223	4303	**6306**	**20711**
	Wheat	21120	15	238	0	4303	**4556**	
	Barley	21130	0	73	0	910	**983**	
	Ryegrass	21140	1798	0	68	704	**2570**	
	Triticale	21150	0	0	0	1777	**1777**	
	Rye	21160	49	167	0	4303	**4519**	
212 - Annual Spring/ Summer Cultures	Corn	21210	1796	194	990	4303	**7283**	**7300**
	Rice	21220	0	0	0	0	**0**	
	Tomato	21230	0	0	0	0	**0**	
	Sunflower	21240	0	0	0	17	**17**	
	Irrigated Area	21201	0	0	0	0	**0**	
213 - Other Agricultural Areas	Vine	22101	0	1000	0	0	**1000**	**8646**
	Orchards	22201	0	0	1000	0	**1000**	
	Olive	22301	0	0	0	0	**0**	
	Pasture	31110	1798	380	165	4303	**6646**	
311 - Cork and Holm Oak	Cork Oak	51101	0	1000	4303	0	**5303**	**10603**
	Holm Oak	51201	0	997	0	4303	**5300**	
312 - Eucalyptus	Adult eucalyptus	51310	1798	285	993	428	**3504**	**22448**
	Eucalyptus AA2017	51301	1798	1000	0	4303	**7101**	
	1 Year Cuts Eucalyptus	51302	1793	1000	991	4303	**8087**	
	Young Cuts Eucalyptus	51303	1798	1000	958	0	**3756**	
313 - Other Hardwoods	Other Hardwoods	51410	1798	975	0	4245	**7018**	**7346**
		51401	0	0	328	0	**328**	
321 - Maritime Pine	Maritime Pine	52110	1798	1000	880	4303	**7981**	**7981**
		52101	0	0	0	0	**0**	
322 - Stone Pine	Stone Pine	52210	0	0	0	16	**16**	**16**
		52201	0	0	0	0	**0**	
323 - Other Softwoods	Other Softwoods	21310	64	0	0	4303	**4367**	**4367**
420 - Spontaneous Herbaceous Vegetation	Agricultural Spont. Herb. Veg.	31201	1370	1000	1000	4303	**7673**	**17273**
	Mountain Spont. Herb. Veg.	31202	0	0	0	4303	**4303**	
	Herbaceous Vegetation AA2017	31203	0	994	0	4303	**5297**	
	Herbaceous Vegetation Cuts	31204	0	0	0	0	**0**	
	Herbaceous Vegetation AA2016	31205	0	0	0	0	**0**	
410 - Bushes	Dense Bushes	61101	1772	1000	995	4298	**8065**	**14467**
	Snow Bushes	61103	0	0	0	0	**0**	
	Bushes AA2017	61104	1785	0	0	4303	**6088**	
		61105	0	0	0	0	**0**	
	Bushes AA2016	61106	314	0	0	0	**314**	
500 - Surfaces without Vegetation	Bare Soil	71110	1782	1000	1000	4303	**8085**	**14652**
	Bare Rock	71201	1798	5	475	4289	**6567**	
610 - Humid Zones	Humid Zones	81110	0	0	0	0	**0**	**0**
620 - Water	Water	91110	1793	1000	998	4303	**8094**	**8094**
Total			**33091**	**17924**	**18364**	**96569**	-	165948

- Based on the dataset analysis, it was possible to conclude that the indexes and metrics were not particularly useful. This is because CNN is capable of extracting meaningful information and features from the images from the raw data without

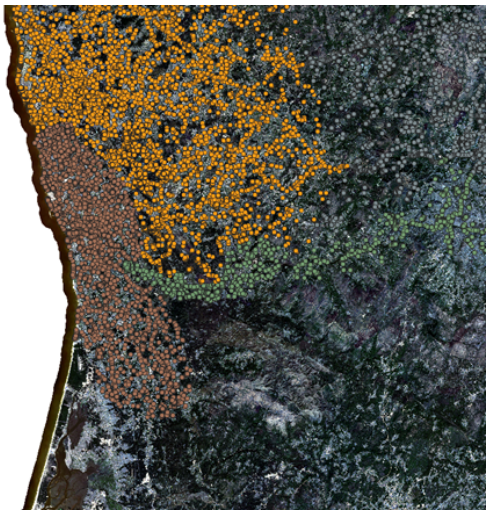
the help of these pre-calculated indexes and metrics. As a result, they were discarded. The elements of the products considered were the ten-band preprocessed GeoTiffs relative to the month of October 2017.

- To align the labels per tile (Figure 3.8a) instead of per LU (Figure 3.8b), it was necessary to cross information between the CSV containing the training points and the GeoTiffs. This change was thought to simplify the use of this data. Essentially, this crossing of information consisted of using the planimetric coordinates present in each label and checking what tile they would fall inside (Figure 3.8). This was made possible due to the characteristics of the GeoTiff files; as raster maps, they allow for the conversion of the pixel coordinates to geographical coordinates (the ones the CSV files possess). The Algorithm 1 shows how this process was done.

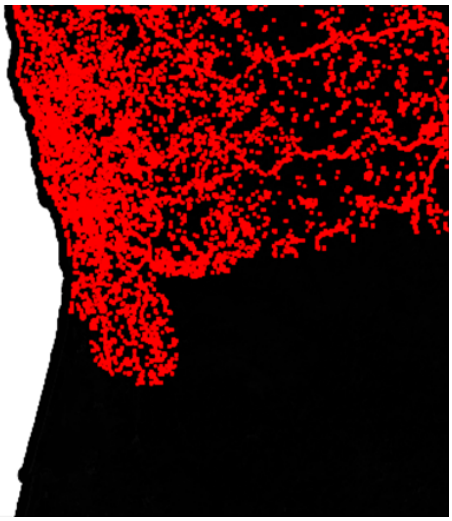
Algorithm 1 Create Labels By Geotiff Instead Of By LU

function SPLIT
▷ This algorithm creates labels for each GeoTIFF tile instead of each LU.
Create a folder called `Labels_by_Geotiff`.
Load the CSV files in the variable `csv_files`.
Load the GeoTIFF tiles in the variable `geotiff_tiles`.
Initialize an empty list called `csv_pixel_index`.
for all data in csv_files **do**
Initialize an empty list called `aux_data`.
for all i, row in data.iterrows() **do**
Get the geo_x, geo_y, and data_class from the row.
for all geotiff_data in geotiff_tiles **do**
Get the bounds of the geotiff_data.
if the geo_x is within the bounds of the geotiff_data and the geo_y is within the bounds of the geotiff_data **then**
Get the pixel_y and pixel_x from the geotiff_data.
Add a new row with the geo_x, geo_y, pixel_x, pixel_y, geotiff_file, and data_class to the `aux_data` list.
Break.
end if
end for
end for
Add the `aux_data` list to the `csv_pixel_index` list.
end for
Concatenate the `csv_pixel_index` list into a DataFrame called `grouped`.
Group the `grouped` DataFrame by the `geotiff_file` column.
Initialize an empty list called `grouped_dataframes`.
for all name, group in grouped **do**
Add the `group.reset_index(drop=True)` to the `grouped_dataframes` list.
end for
for all aux, data in enumerate(grouped_dataframes) **do**
Get the filename from the data.
Remove the `.csv` extension from the filename.
Remove the last 6 characters from the filename.
Get the filepath as `os.path.join(labels_folder, filename + '.csv')`.
Save the data to the filepath.
end for
end function

- The originally thought-out concept was based on the assumption that a large number of contiguous pixels were labeled; however, this was not the case. So, the idea of doing image segmentation while using CNN for pixel classification



(A) Overlay of the training points over one GeoTiff previous training points per tile



(B) Overlay of the training points over one GeoTiff after training points per tile

FIGURE 3.8. Overlay of Tile with training points by LU and by tile

was dropped. Considering that the percentage of labeled pixels per tile was around (0.02294%) considering the total number of pixels in a tile (10980x10980), pursuing the segmentation idea was not feasible. The new idea consisted of using the known location of the labeled pixels of the tiles, previously split by tile, and using them to crop the tile into smaller patches of land. These patches would be centered in the known location and would consider all the surrounding areas to be of the same class as the central pixel.

- To apply this cropping strategy, it was necessary to load the training points per tile file, convert each label location to a pixel location, and then retrieve the center point of the patches that needed to be created using this information.
- The next step involved cropping the new GeoTiff based on the center point location information retrieved. Each image patch would retain all data from the Sentinel-2 product, keeping information from every band. We chose a patch size of 33x33 pixels, which corresponds to an area of 108,900 m2 at a spatial resolution of 10 meters. This crop is shown in Figure 3.9b. It was important to consider that the use of 33x33 patches required the center point (used for labeling) to be at least 17 pixels away from the tile's edge. If this condition wasn't met, the cropping would be skipped to avoid falling outside the image. In the full dataset, this situation occurred 390 times, resulting in a reduction of training points from 165,948 to 165,558, as indicated in Table 3.8.

The choice of the size of the patch was made while trying to strike a balance between the image size needed to implement a CNN architecture; something very small won't have enough size to be subject to convolution and pooling. On the contrary, with the increase in size and considering that the classification is

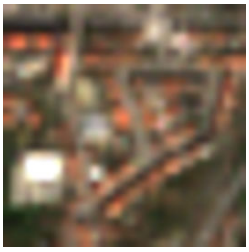
Algorithm 2 Data Crop

```
function CROP
        ▷ This algorithm creates labels for each GeoTIFF tile instead of each LU.
    Read or create folders to store the new dataset
    for all data in dataset_folder_path do
        Create a folder for data
    end for
    Read CSV files with pixel labels
    Read paths of GeoTIFF tiles
    Define the window size for cropping
    Initialize an empty list for failed rows
    for all CSV label files do
        for all GeoTIFF tile files do
            if CSV file matches GeoTIFF tile then
                for all rows in CSV data do
                    Extract pixel coordinates and class
                    Determine the output folder
                    Generate an output file name
                    Crop around the pixel from the GeoTIFF tile
                    if crop is successful then
                        Save the cropped image
                    else
                        Add the row to the list of failed rows
                    end if
                end for
            end if
        end for
    end for
    Save the list of failed rows to a CSV file
    Read the paths of data folders
    Split data into training, validation, and test sets
    Save the file paths for each set to text files
end function
```

attributed to the image, other classes in the image may not be considered. For example, if the point is in the middle of a forest or mountain, it probably won't be crossed by anything else. However, if this same forest point refers to an edge of the forest, other classes like artificialized (i.e., roads that may exist) won't be detected. These crops were stored in a structure like the COSsim level three nomenclature shown in Table 3.5. This choice was made to decrease the number of classes, especially considering that some of the classes are poorly represented. So, the labels were converted from the training point class to level three of the COSsim, and then the crops were stored inside a general file for each class, with the original name of the tile plus the class and the index in which it was cropped. From this point on, the original tile didn't matter. Besides this, from the whole universe of data, only one month's worth of data was used from each different zone (October 2017). This process ocurred as shown in the Algorithm 2

The files that were indexes, metrics, and the rest of the months weren't considered. The first two already explained why the question surrounding the choice of only one month from the data from each of the six months was due to the apparent lack of significant changes from month to month. The Figure 3.9 provides a comparison between the tile (Figure 3.9a) and a cropped patch (Figure 3.9b).

(A) 29TNF Tile



(B) 29TNF Artificialized Patch Example

FIGURE 3.9. 29TNF tile and one training point in RGB

- With the data now split into smaller patches and stored by class, the main goal changed; it was now to get the data ready to be inputted into the CNN. This would mean, among other things, that it had to be divided into training (70%) with a total of 115884 samples, testing (10%) with 16568 samples, and validation data (20%) with 33106 patches. The distribution across each of the Level 3 classes, as well as some other information, can be seen in Table 3.8 and in a chart format in Figure3.10.

TABLE 3.8. Dataset Division into Train, Validation, and Test

Level 3	Total/level	Training	Validation	Testing
100 - Artificialized	21,857	15,299	4,371	2,187
211 - Annual Autumn/Winter Cultures	20,684	14,478	4,136	2,070
212 - Annual Spring/Summer Cultures	7,297	5,107	1,459	731
213 - Other Agricultural Areas	8,587	6,010	1,717	860
311 - Cork and Holm Oak	10,603	7,422	2,120	1,061
312 - Eucalyptus	22,445	15,711	4,489	2,245
313 - Other Hardwoods	7,330	5,131	1,466	733
321 - Maritime Pine	7,943	5,560	1,588	795
322 - Stone Pine	16	11	3	2
323 - Other Softwoods	4,367	3,056	873	438
420 - Spontaneous Herbaceous Vegetation	17,250	12,075	3,450	1,725
410 - Bushes	14,444	10,110	2,888	1,446
500 - Surfaces without Vegetation	14,642	10,249	2,928	1,465
610 - Humid Zones	0	0	0	0
620 - Water	8,093	5,665	1,618	810
Total	**165,558**	**115,884**	**33,106**	**16,568**

- This was done randomly for each class, keeping the same percentage for each class. This implies that, even though the number of samples per class is not

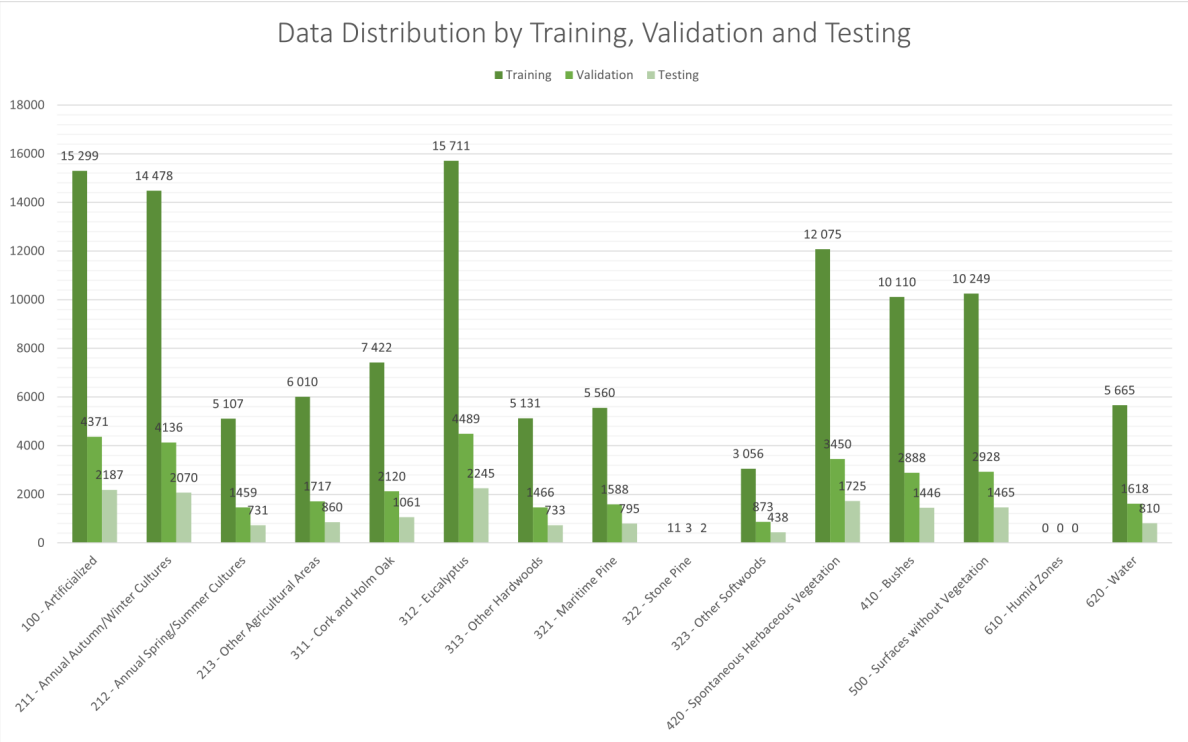


FIGURE 3.10. Chart with the distribution of the data per Training, Validation and Testing

equal due to the non-normal distribution, the proportion of samples allocated to each of the groups within each class remains constant.

- After the data had already been divided into three parts (testing, validation, and training), the GeoTiffs were reloaded in the subsequent step. Each file was then normalized so that all the pixel reflectance values would range from 0 to 1. Then, each cropped GeoTiff file was saved in a folder corresponding to their final goal: test, train, and validate. However, they weren't saved as GeoTiffs but as NumPy array files (npy, these files created by the Python library NumPy are essentially a more compact and efficient way of storing arrays than the format being used). Of each file, two versions were saved: a ten-band version like the provided Sentinel-2 product and a three-band RGB file. This choice was made because of two different factors: the first was that the performance could be measured from different perspectives; the second was the difference between using all of the information and only considering a lesser number of bands; and the third was the interest in trying to use transfer learning. Transfer learning involves adapting pre-trained CNN models to new data. These algorithms expect data with certain characteristics depending on the model, typically RGB. Which gave an even greater incentive. For a better comprehension of the whole process, the Algorithm 3 gives a pretty good overview.
- After the entire dataset had been divided into three folders and was ready to be fed into the CNN, the only thing missing was a label file for each folder, given

Algorithm 3 Data Normalization and .npy storing

```
function DataNormalizationAndReady
                        ▷ This algorithm normalizes and stores data in .npy files.
    Load previously created dataset_dir
    Initialize classes with class names
    Initialize data_folders and data_splits
    for all i, folder in data_folders do
        Read paths from data_splits using index i
        Read files from dataset_dir using read_file_paths()
        Convert geotiffs to arrays using convert_geotiffs_to_arrays()
        Create dataset_path using os.path.join()
        Extract labels using extract_labels()
        Save labels as folder_labels.npy
        Read files from dataset_path
        Create RGB data folder and paths using folder_creator()
and os.path.join()
        for all file in data_files do
            Load and select bands from file
            Normalize the data
            Save as name_RGB.npy
        end for
    end for
end function
```

that the training would be performed using one-hot encoding. This means that the following characteristics are present in a matrix similar to Table 3.9:

Table 3.9. One-Hot Encoding Label Example

	0	1	2	3	...	Nº Classes -1
1	1	0	0	0	...	
2	1	0	0	0	...	
3	0	1	0	0	...	
4	0	0	1	0	...	
5	0	0	0	1	...	
...	...	...	...	...	...	
N-1 Samples						

Each column in the Table 3.9 represents a class, and each line represents a certain label. So in the combination of row and column where the is a one, it symbolizes that that given label is from the class that corresponds to the column.

When all this was completed, the next step was to develop the CNN model.

3.4. Convolutional Neuronal Network Architecture

In what concerns the CNN architecture employed, a division can be made to distinguish what was tried:

- custom models using 10 bands as input
- custom models using RGB bands as input
- tranfer learning-based models using RGB bands

All the architectures were defined on the before-mentioned computer using Python 3.11 and the Keras library with the Keras backend.

3.4.1. Sentinel-2 Products Model

The architecture of CNN used when considering the full ten bands is the one that retains the native spectral information from the provided data, given that the spatial component has been changed, using patches of dimensions 33x33x10 instead of the tiles with dimensions 10980x10980x10. This neuronal network architecture design (Figure 3.11) with the goal of image classification has a repeating block structure due to the reduced image size (33x33 pixels), which makes it unsuitable to use many layers and makes it not a very deep network. The goal was to achieve a balance between computational complexity and performance.

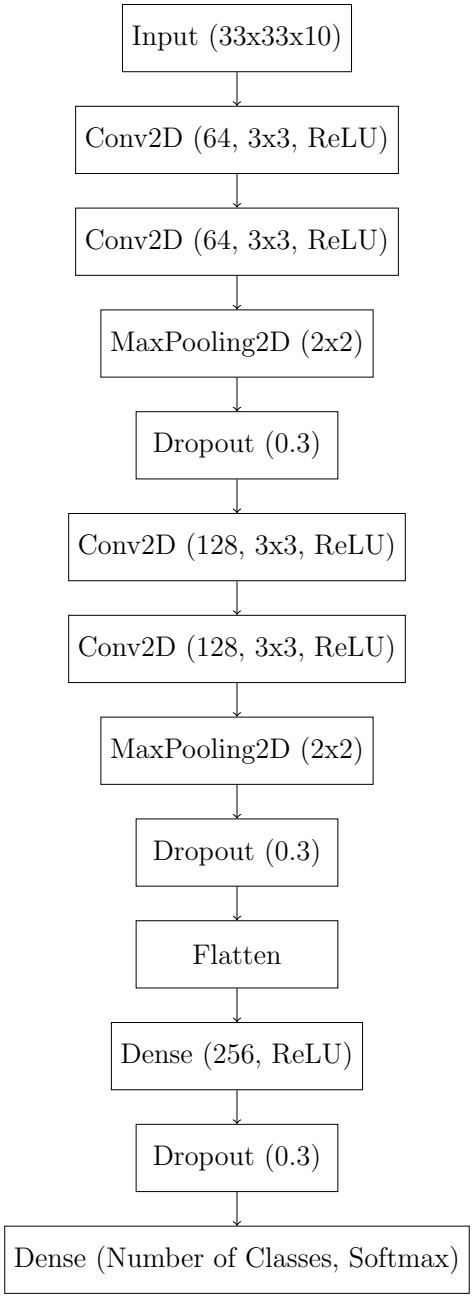


FIGURE 3.11. 10 Bands Model Architecture

This referred block is composed of two stacked convolutional layers, which have a number of filters equal to 64 in the first convolutional layer and 128 in the second. These filters have a size of 3x3 pixels. The activation function of this block is the ReLu. After the convolutional layers, a MaxPooling layer with a window size of 2x2 is applied, reducing the feature map to half of its previous size. After the MaxPooling layer, a Dropout layer with a dropout ratio of 20% is added to diminish the chances of overfitting and allow the network to generalize new data. After this is repeated two times, a flatten layer converts the feature maps that result from the last block to a one-dimensional array. Then this array goes through a dense layer with 256 units that use the ReLu activation function, followed by a new dropout layer, this time with a rate of 30%. The final layer of this architecture is a dense layer, with the number of units equal to the number of classes. The activation function here used is the softmax, which will produce the class probabilities.

3.4.2. RGB Model

When considering the RGB data only the three corresponding bands were used (instead of ten bands). The defined architecture is close to the one described in the previous section, but only three input channels are used, as shown in Figure 3.12.

Other architectures were attempted; through them, the increase and decrease, for example, in convolutional layers occurred; however, these are the architectures that got the best results.

3.4.3. Transfer Learning

Besides the customized architectures described in the previous sections, additional architectures were defined, based on the idea on the idea of transfer learning. The two tested transfer learning-based architectures were the MobileNet [**50**] and the EfficientNet [**51**].

3.4.3.1. *MobileNet* The first attempt at transfer learning was with a focus on MobileNetV3. This CNN comprises two different versions, the small and large variants. These two models present differences in what concerns their structure, with the larger being the most complex and having a superior number of hidden layers, as evident by their names. The structure employed in the tests for both the large and the small is the same and is shown in the Figure 3.13.

There are, however, some requirements that need to be met for these networks to operate. The data values should range from 0 to 255, and the minimum input dimension is 32x32 (ideally it should be at least 224x224).

In the first part, the data failed to comply as it was normalized between 0 and 1, so it had to be multiplied by 255. In terms of the data dimensions, they were kept, as they are above the minimum input size, and the resizing test that was done didn't achieve better results. Only a drastic increase in the training time would make the increase in data size so that the data would be loaded, batched, and shuffled in parts.

With the data now ready to be loaded into the CNN, the dimensions stayed at 33x33x3, with pixel values ranging between 0 and 255. This model was loaded with its pre-trained

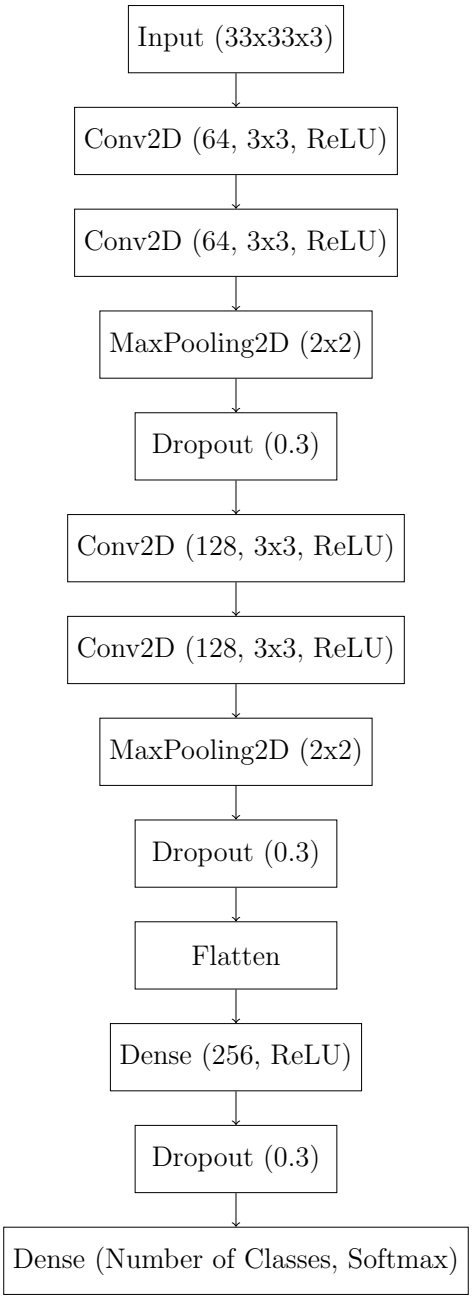


FIGURE 3.12. RGB Model Architecture

weights from the ImageNet dataset [**19**]. To this model, and as expected in transfer learning, other layers were added—in this case, four. A first layer that flattens the data, a dense layer with 256 units and ReLu activation, a dropout layer with a 30% dropout rate, and a final dense layer that is the same as the one used in the other architectures, along with a SoftMax function to convert to the land classification classes

Important to this architecture's implementation (transfer learning) was the freezing of the pre-trained weights of each layer and neuron relative to ImageNet training. This ensures that the layers remain static during training and that the only layers that are trained are the deeper ones. As part of the transfer models' fine-tuning, it was determined how many layers would remain frozen with their pre-trained weights and how many layers

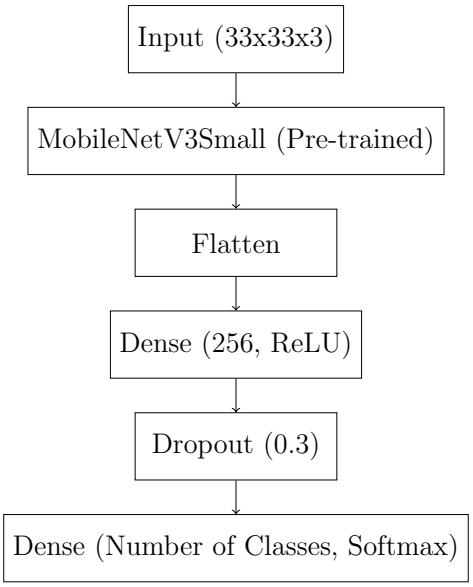


FIGURE 3.13. Transfer Learning MobileNet Model Architecture

would be trained. This was accomplished by starting with all model layers frozen and removing the freeze one layer at a time, beginning with the deepest layers, all while evaluating the performance metrics. With this process in mind, the number of unfrozen layers was seven in the case of MobileNetV3.

3.4.3.2. *EfficientNet* The EfficientNet architecture (Figure 3.14), like MobileNet (Figure 3.13), is comprised of various versions. Instead of the large or small models, EfficientNet has models that go from B0, the less complex, to B6, the most complex alternative. The more complex versions, as in the case of the MobileNet, are the ones that promise better results; however, they are also the ones that are more demanding on the hardware and require a larger dataset for an adequate training process.

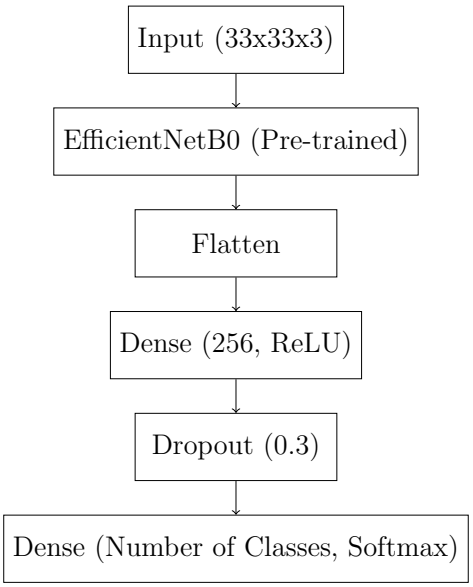


FIGURE 3.14. Transfer Learning EfficientNet Model Architecture

The complete architecture is similar to the one based on MobileNet previously described: the size of the patches used in the input is 33x33x3, the pre-trained model used in this case is EfficientNetB0, and the remaining network layers are the same as previously described. The number of unfrozen layers in this case was five.

CHAPTER 4

Experimental Results

This chapter provides an overview of the tests conducted and an explanation of the results obtained during this stage.

In terms of the hardware and platform used for training and validation of the CNN, it was an NVIDIA T1000 with 4 GB of VRAM along with a 6^{th} generation i7 processor with 64 GB of RAM. This hardware will be one of the most limiting factors in the training; the maximum memory the computer has will, for example, limit the amount of data that can be input to the network, influencing many parameters like batch size (the number of samples of the data propagated by the network in each iteration).

4.1. Performance Metrics

Several metrics were used to evaluate the performance of the developed CNN architectures and how they behaved during training. These were implemented throughout the training and validation phases. Accuracy and loss were used for both the training and validation datasets, along with a plot that provided more user-friendly information on how the different values of loss and accuracy evolved during the training process (epochs), also known as the learning curve. Precision, recall, F1-score, and the use of a confusion matrix were used as performance measures when testing the trained network networks on test sets (containing different images than the ones used during training).

As a summary of each metric, it can be said that accuracy refers to the proportion of correctly classified data elements, while loss quantifies the difference between the predicted and actual probabilities of the label's value. It is important to note, however, that these performance indicators work better on balanced datasets, which the dataset used here does not. So, it is better to use the other mentioned performance indicators. To better understand the way these performance indicators are obtained, it is necessary to first define four concepts: false positives, false negatives, true positives, and true negatives. False positives happen when the model incorrectly predicts a positive class when it should be negative. Meaning that the architecture incorrectly identified it as belonging to the class of study. A false negative is the opposite of a true positive, as the name implies; the model predicts something as not belonging to the class when in reality it belonged there and hence should have been positive. True positive and true negative refers to instances in which the architecture correctly predicts that a sample belongs to or does not belong to a given class. With these concepts now explained, the remaining metrics can be better introduced. Precision relates to the ratio of true positive predictions to all

positive predictions, both true and false. Its formula is:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{4.1}$$

Essentially calculates the accuracy of the positives in the model. The recall has the formula:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{4.2}$$

, measuring the ratio and the number of actual positive instances that are correctly predicted as positive. The F1-score relates both, combining them to make a more balanced metric. It has the formula:

$$\text{F1 Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.3}$$

In conclusion, precision aims to minimize the number of false positives and the recall of false negatives, whereas the F1 score allows for an attempt to find a balance between the two. The confusion matrix is a tool that enables more direct data collection, providing a summary of the number of true and false positives as well as the number of true and false negatives, which allows a calculation of the various metrics discussed previously.

4.2. Experimental Findings

There were two distinct steps in this testing and implementation phase of the previously defined and explained models.

The first one, which is the training and validation of the network, as the name implies, uses the training and validation datasets created before, and this is the part that is more time-consuming and computationally demanding. In this first part, accuracy and loss are evaluated and the learning curve is presented, which allows for a better understanding of how the training progressed.

The second section is the testing section, which utilizes the testing portion of the dataset. Consisting of the loading of previously trained models, followed by data classification using them. Precision, recall, and F1 score are the criteria for this section of testing. The confusion matrix serves as a graphical aid here. This second element will be as significant as the first, if not more so, due to the uneven distribution of data between classes.

4.2.1. Training and Validation

In this, initial, and typically most time-consuming phase, of implementing CNNs, when dealing with already preprocessed data, one thing must be completed before anything else. This is the loading of the already-divided data from the .npy files and the corresponding labels. The loading of the data can be better understood with the following algorithm; however, a brief description would include loading the data relative to the training points, loading the labels, and constructing a dataset using the TensorFlow framework.

By permitting the creation of a dataset-type variable, this framework facilitates the shuffling and batching of data before feeding it to the CNN, as well as the multiplication

Algorithm 4 Data Loading

```
function LoadAndPreprocessData
    Initialize dataset_dir to 'D:/newDataV2'
    Initialize folder_paths with ['RGB_train', 'RGB_test', 'RGB_validation']
    Initialize labels_files with
['train_labels.npy', 'test_labels.npy', 'validation_labels.npy']
    Initialize input_shape to (33, 33, 3)
    Initialize batch_size to 128
    for all file_name in labels_files do
        Load labels from os.path.join(dataset_dir, file_name)
        Convert labels to TensorFlow tensors with data type tf.float32
    end for
    for all folder_path in folder_paths do
        Read file paths from
os.listdir(os.path.join(dataset_dir, folder_path))
        Sort file paths alphabetically
        Initialize empty lists training_data and validation_data
        for all file_name in file_paths do
            Load data from os.path.join(dataset_dir, folder_path, file_name)
            Append data to training_data if in training folder
or validation_data if in validation folder
        end for
        Stack training_data and validation_data to create tensors
        Create TensorFlow datasets from tensors
        Map function to normalize data in datasets
        Shuffle and batch datasets
    end for
end function
```

by 255 to make it between 0 and 255 and the resizing of images. This also means that if the data itself or after resizing is too large for the available computer memory, it will be loaded in pieces so that the system can manage it. Anything other than shuffling and batching must be performed during the testing phase, as the training and validation data must have the same characteristics as the testing data.

Table 4.1 provides an overview of the results of the training and validation of each distinct architecture with the previously mentioned data. These are the results of training and validating the data with the above-mentioned architectures (3.4.2), which yielded the best results after many experiments.

TABLE 4.1. Comparison of the Training Performance between the CNNs used

	Models			
Metric	10 Band	RGB	MobileNetV3	EfficientNetB0
Epochs	22	26	29	23
Average Time per Epoch (s)	3311.15	1168.59	315.80	948.26
Accuracy (%)	97.51	96.96	95.10	91.19
Val_Accuracy (%)	98.51	97.84	89.67	92.96
Loss	0.0733	0.0891	0.1403	0.2574
Val_Loss	0.0527	0.0795	0.3733	0.2452

By analyzing this table, it is easy to see that the architectures that were designed from scratch for this purpose performed significantly better in all metrics, achieving significantly

lower losses and significantly higher values of accuracy. In contrast, the time required to build the network is also very large. The average time per epoch required to train the 10-band architecture versus MobileNetV3 demonstrates the greatest difference. Taking into account the number of epochs (22 for the 10 Band and 29 for the MobileNetV3), this difference amounts to a total of 63687,1 s (17 hours, 41 minutes, and 27 seconds).

Comparing each of the architecture types, that is, transfer learning and those defined from scratch, it is possible to observe that the 10-band model performs marginally better than the RGB model. The difference in precision is negligible (10 bands = 97.51 and RGB = 96.96 for training and validation, 98.51 and 97.84, respectively). The difference is greater when considering the loss values, particularly the validation loss of 0.0527 for the 10 bands and 0.0795 for the RGB model (during training, the difference was smaller, 0.0733 for the 10 bands and 0.0805 for the RGB). Taking into account the number of epochs and the average time of each epoch, it is clear that the 10-band model requires approximately three times as much time to train and validate (1,168.59 s vs. 3,311.15 s); however, it requires fewer epochs, 22 as opposed to 26.

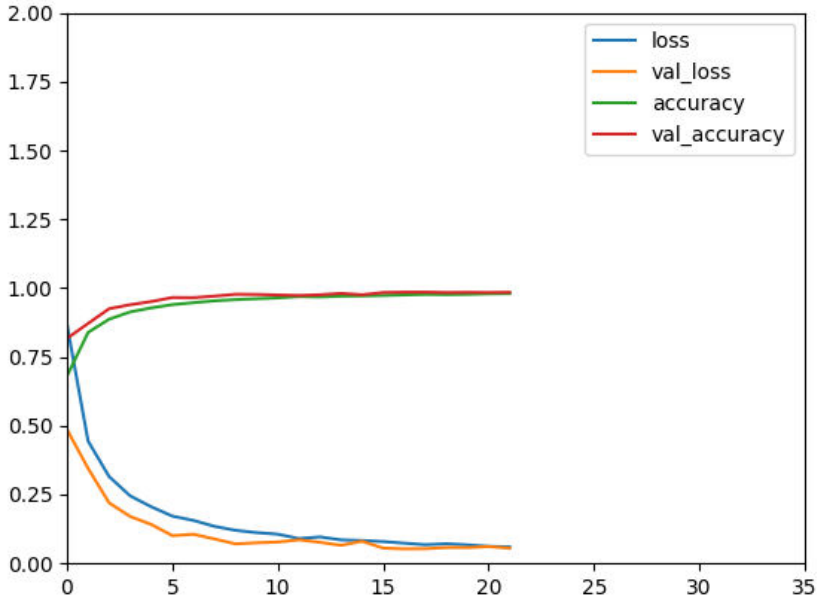


FIGURE 4.1. Learning Curve of the 10 Band CNN

The models learning curves (Figure 4.1, Figure 4.2, Figure 4.3, Figure 4.4) illustrate the progression of learning and validation across epochs.

Observing them confirms what was previously stated, namely that custom architectures perform better, are more stable during training, have fewer fluctuations, and converge more quickly.

As with the other parameters, the 10-band model (Figure 4.1) appears slightly superior to its 3-band (Figure 4.2) counterpart.

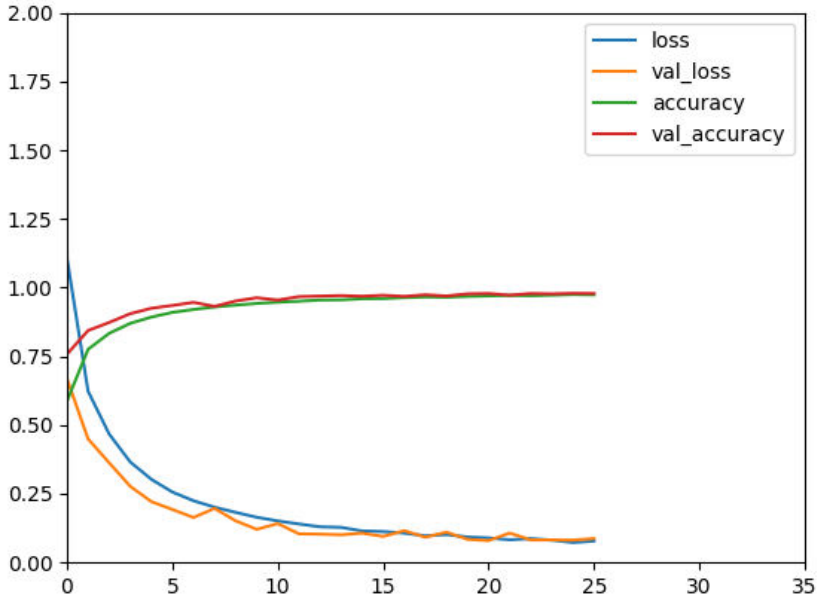


FIGURE 4.2. Learning Curve of the CNN for RGB

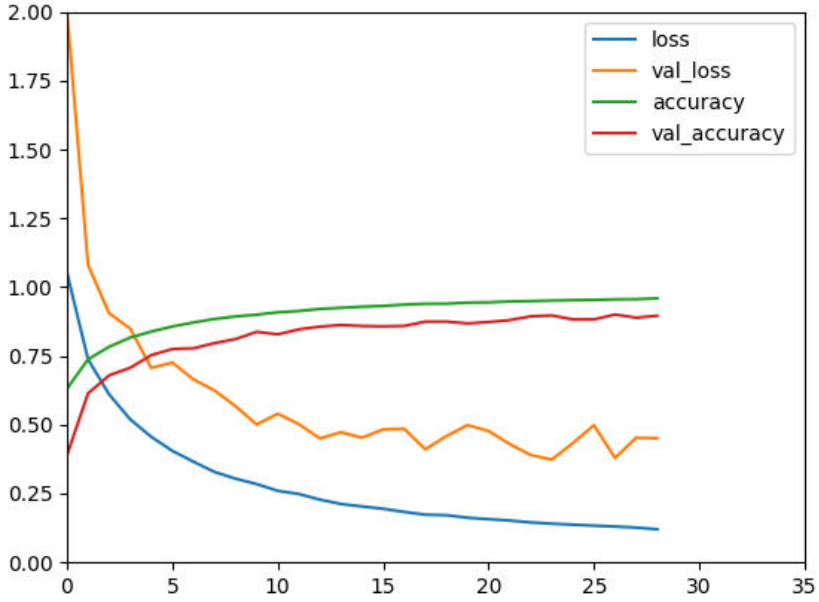


FIGURE 4.3. Learning Curve of the MobileNetV3 Transfer Learning Architecture

The validation loss line is more rounded, and there is not a single spike in the validation accuracy. Specifically, the loss validation parameter in EfficientNetB0 (Figure 4.4) demonstrates significantly larger fluctuations for the transfer learning models. EfficientNetB0 is the only model among the two transfer learning models that exhibits greater convergence

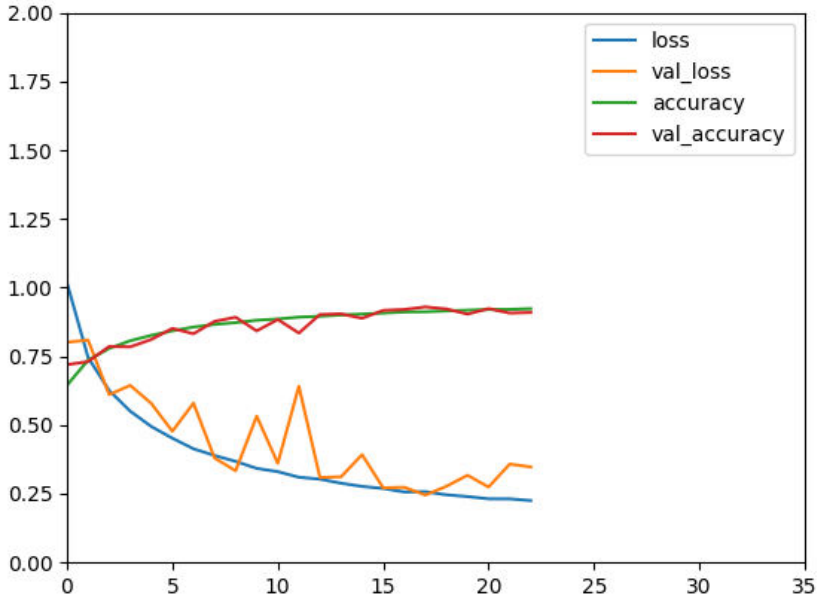


FIGURE 4.4. Learning Curve of the EfficientNetB0 Transfer Learning Architecture

between the training and validation data, despite exhibiting the most fluctuations in the loss validation. MobileNetV3 (Figure 4.3) is the architecture that demonstrates superior convergence performance. Among the four distinct models, the one whose training and validation lines are separated by a greater distance.

Unmentioned factors must be taken into account concerning these two architectures that rely on previously trained models. After testing, the learning rates for both architectures were adjusted to 0.0152 because it was determined that this value would produce the best results. This was necessary because these networks performed significantly worse than their personalized counterparts. Aside from this, after the networks were loaded with the weights developed when they were trained with the Imagenet dataset, these layers were largely "frozen", meaning that these weights were maintained, and the only layers that changed during training were those added here. However, as with the need to add the learning rate through testing and make some of the last layers available for training, it was possible to improve on the results until a certain point, past which the results would once again degrade. This quantity of unfrozen layers would be seven for MobileNetV3 and five for EfficientNetB0.

4.2.2. Testing

In this final phase of the analysis of the results, the focus is on the testing of the trained models; this is the step that is closer to what a user of the models will experience. In this phase, the performance of these models is studied through the evaluation of the performance metrics of the tests, precision, recall, and F1-score.

In this sense, four different tables are presented, each accompanied by its confusion matrix. These tables represent each of the four architectures:

- 10-band CNN: Table 4.2 and confusion matrix in the Figure 4.5
- 3-band CNN: Table 4.3 and confusion matrix in the Figure 4.6
- MobileNetV3 based CNN: Table 4.4 and confusion matrix in the Figure 4.7
- EfficientNetB0 based CNN: Table 4.5 and confusion matrix in the Figure 4.8

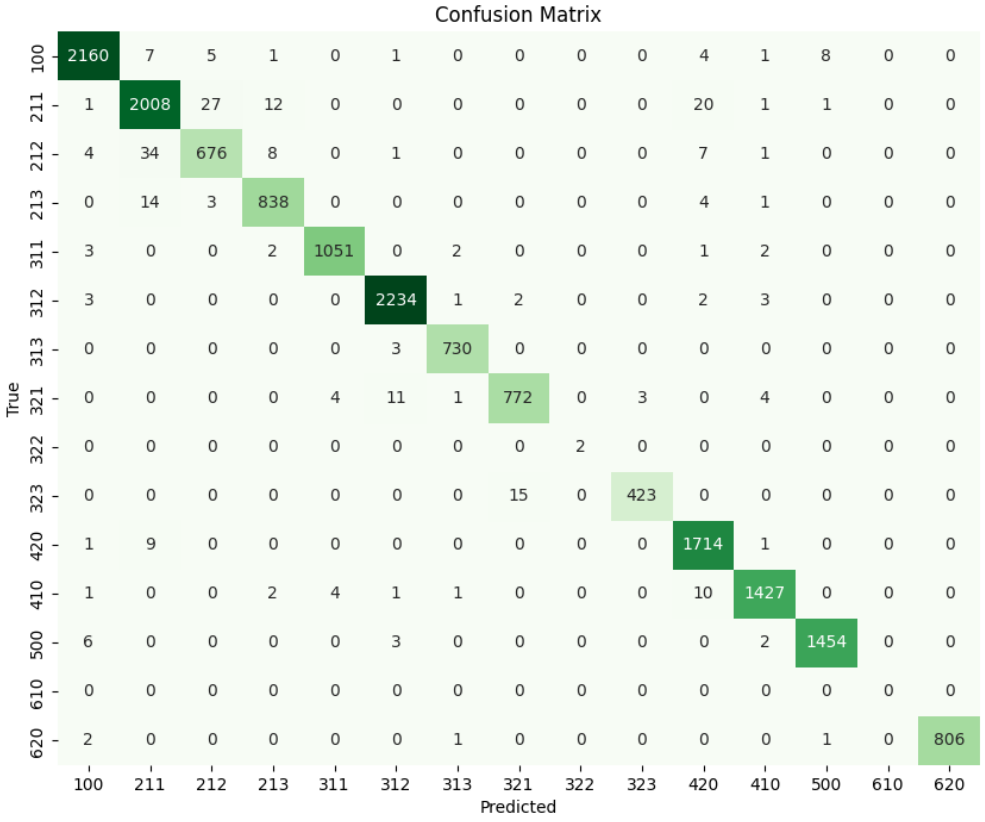


FIGURE 4.5. Confusion Matrix of the 10 Band CNN

Analyzing Table 4.2 and its associated confusion matrix (Figure 4.5) for the 10-band CNN model reveals that it performs worse with seasonal cultures, agricultural areas, and spontaneous herbaceous vegetation classes. It is also possible to see that it performs perfectly in the two classes: water and stone pine. For these classes, the F1-score is 1. Considering the whole data, the fact that precision and recall are mostly similar or close means that the balance between false positives and false negatives is good.

The evaluation is mostly done considering the F1-score, as it takes into consideration both precision and recall. The 'Support' column indicates the sample count for the specific class, making it possible to check the data imbalance; for example, there are no samples of humid zones or a very low amount of stone pine samples.

TABLE 4.2. Performance Metrics from 10 Band CNN in Testing Data

COSsim Level 3 Class	Precision	Recall	F1-Score	Support
100 - Artificialized	0.99	0.99	0.99	2187
211 - Annual Autumn/Winter Cultures	0.97	0.97	0.97	2070
212 - Annual Spring/Summer Cultures	0.95	0.92	0.94	731
213 - Other Agricultural Areas	0.97	0.97	0.97	860
311 - Cork and Holm Oak	0.99	0.99	0.99	1061
312 - Eucalyptus	0.99	1.00	0.99	2245
313 - Other Hardwoods	0.99	1.00	0.99	733
321 - Maritime Pine	0.98	0.97	0.97	795
322 - Stone Pine	1.00	1.00	1.00	2
323 - Other Softwoods	0.99	0.97	0.98	438
420 - Spontaneous Herbaceous Vegetation	0.97	0.99	0.98	1725
410 - Bushes	0.99	0.99	0.99	1446
500 - Surfaces without Vegetation	0.99	0.99	0.99	1465
620 - Water	1.00	1.00	1.00	810

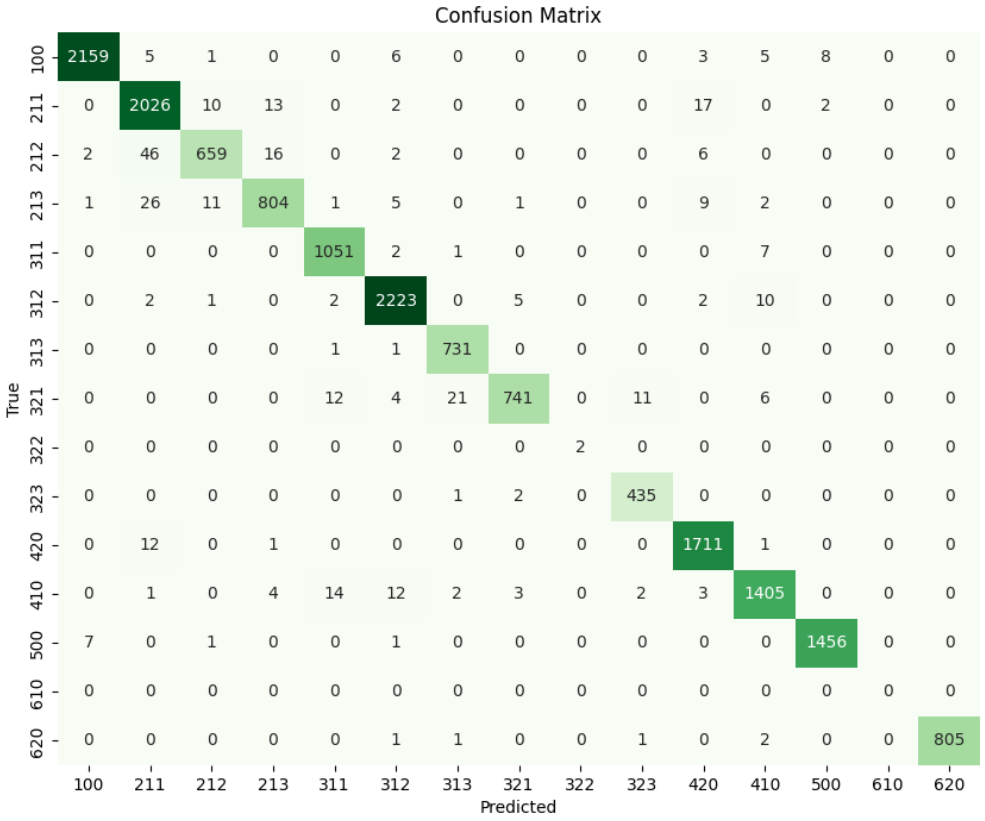


FIGURE 4.6. Confusion Matrix of the RGB CNN

When considering the RGB model, both the table (Table 4.3) with class-specific metrics and the confusion matrix (Figure 4.6) consistently show high precision, recall, and

f1-score across the classes. Although marginally lower than the 10-band model (especially in the worst classes), it is still a model worth considering; it is less computationally demanding, takes less time to train, and the data needed to use it is simpler.

Table 4.3. Performance Metrics from RGB CNN in Testing Data

COSsim Level 3 Class	Precision	Recall	F1-Score	Support
100 - Artificialized	1.00	0.99	0.99	2187
211 - Annual Autumn/Winter Cultures	0.96	0.98	0.97	2070
212 - Annual Spring/Summer Cultures	0.96	0.90	0.93	731
213 - Other Agricultural Areas	0.96	0.93	0.95	860
311 - Cork and Holm Oak	0.97	0.99	0.98	1061
312 - Eucalyptus	0.98	0.99	0.99	2245
313 - Other Hardwoods	0.97	1.00	0.98	733
321 - Maritime Pine	0.99	0.93	0.96	795
322 - Stone Pine	1.00	1.00	1.00	2
323 - Other Softwoods	0.97	0.99	0.98	438
420 - Spontaneous Herbaceous Vegetation	0.98	0.99	0.98	1725
410 - Bushes	0.98	0.97	0.97	1446
500 - Surfaces without Vegetation	0.99	0.99	0.99	1465
620 - Water	1.00	0.99	1.00	810

After analyzing both the training and validation phases, it is not surprising to find that the performance of the two above-mentioned models (custom design models) is superior when compared to the transfer learning-based models. However, in the end, their performance metrics aren't all that bad. F1-score, precision, and recall consistently exceed 0.9 for all classes.

Table 4.4. Performance Metrics from MobileNetV3 Transfer Learning in Testing Data

COSsim Level 3 Class	Precision	Recall	F1-Score	Support
100 - Artificialized	0.98	0.95	0.96	2187
211 - Annual Autumn/Winter Cultures	0.89	0.96	0.92	2070
212 - Annual Spring/Summer Cultures	0.88	0.79	0.83	731
213 - Other Agricultural Areas	0.90	0.89	0.89	860
311 - Cork and Holm Oak	0.91	0.95	0.93	1061
312 - Eucalyptus	0.91	0.95	0.93	2245
313 - Other Hardwoods	0.90	0.95	0.92	733
321 - Maritime Pine	0.90	0.78	0.83	795
322 - Stone Pine	1.00	1.00	1.00	2
323 - Other Softwoods	0.90	0.99	0.94	438
420 - Spontaneous Herbaceous Vegetation	0.98	0.95	0.96	1725
410 - Bushes	0.92	0.84	0.88	1446
500 - Surfaces without Vegetation	0.96	0.99	0.98	1465
620 - Water	0.98	0.98	0.98	810

When comparing the transfer learning-based models with each other, it is possible to check that the performances of both are pretty similar to one another; in some classes, one is better than the other in what concerns precision, but then the recall evens out.

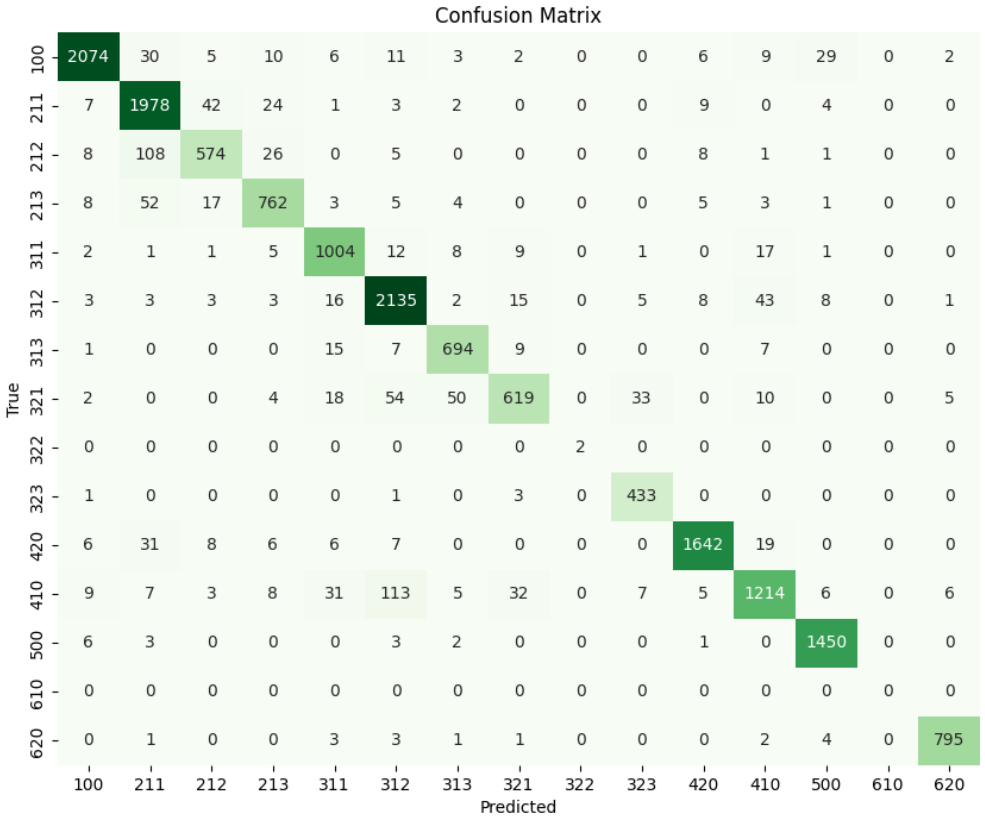


FIGURE 4.7. Confusion Matrix of the MobileNetV3 Transfer Learning Architecture

This is seen in the F1-score, where they have the same value in all the classes. The class in which the performance gets the worst results in all of the tables is the recall in the MobileNet-based model with 0.78. But even this minimum is pretty much equal to the recall in the same class from the EfficientNet-based model with 0.79.

After conducting the tests, and as an overview, some things can be said. It became evident that the custom architectures designed specifically for this task consistently outperformed their transfer-learning-based alternatives. Of these, the one that obtains the best performance is the 10-band CNN. The RGB model appears pretty similar, only slightly below in terms of precision and recall over the classes. Due to this, and with the reduction in the general size of the data it needs, it is still a viable alternative.

In what concerns the transfer learning-based model, despite, as stated, performing worse in comparison, it still shows competitive performance metrics. Between the two architectures, as in the custom model case, the differences are minimal, which indicates comparable effectiveness in the context of their use.

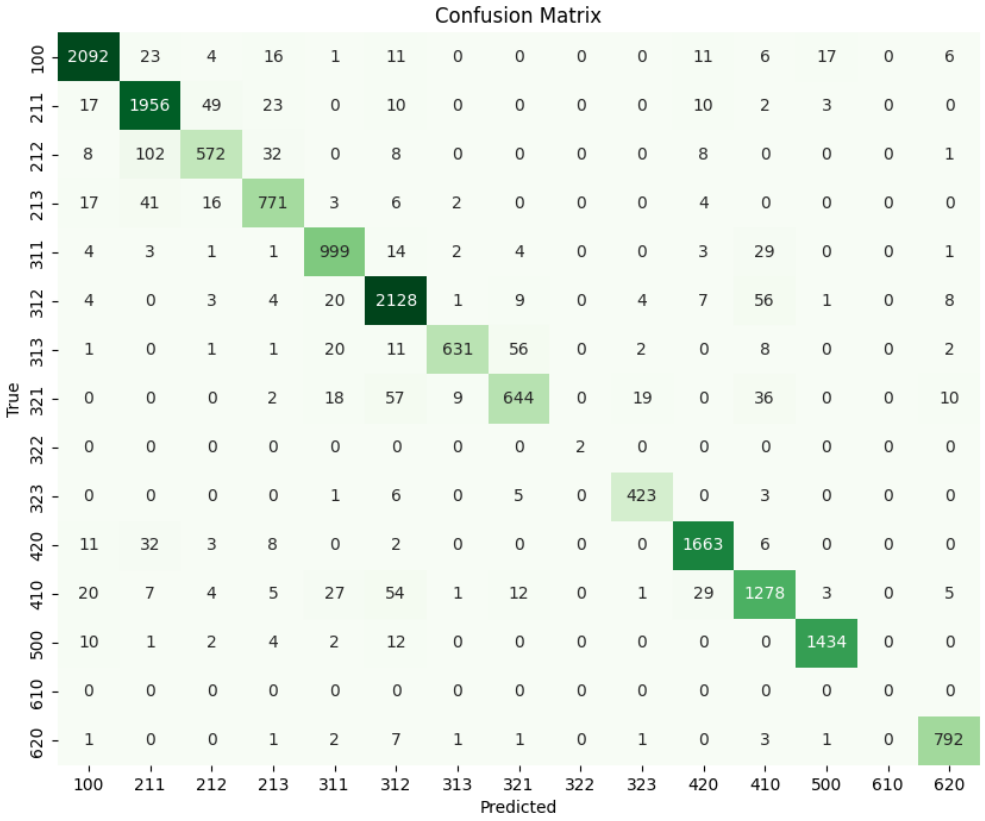


FIGURE 4.8. Confusion Matrix of the EfficientNetB0 Transfer Learning Architecture

TABLE 4.5. Performance Metrics from EfficientNetB0 Transfer Learning in Testing Data

COSsim Level 3 Class	Precision	Recall	F1-Score	Support
100 - Artificialized	0.96	0.96	0.96	2187
211 - Annual Autumn/Winter Cultures	0.90	0.94	0.92	2070
212 - Annual Spring/Summer Cultures	0.87	0.78	0.83	731
213 - Other Agricultural Areas	0.89	0.90	0.89	860
311 - Cork and Holm Oak	0.91	0.94	0.93	1061
312 - Eucalyptus	0.91	0.95	0.93	2245
313 - Other Hardwoods	0.98	0.86	0.91	733
321 - Maritime Pine	0.88	0.81	0.84	795
322 - Stone Pine	1.00	1.00	1.00	2
323 - Other Softwoods	0.94	0.97	0.95	438
420 - Spontaneous Herbaceous Vegetation	0.96	0.96	0.96	1725
410 - Bushes	0.90	0.88	0.89	1446
500 - Surfaces without Vegetation	0.98	0.98	0.98	1465
620 - Water	0.96	0.98	0.97	810

CHAPTER 5

Conclusions and Future Work

This thesis examined the viability of using CNNs to classify land cover using high-resolution remote sensing imagery. With this objective in mind, Sentinel-2 mission data (MSI data from October 2017 to September 2018) was preprocessed in the THEIAS before being downloaded and preprocessed by the DGT, along with training points for zones within the imagery. Before this data could be utilized, it had to be processed, and deep learning models had to be developed and refined.

This chapter will provide an overview of the entire body of work and answer the questions posed in Chapter 2. In addition, future work or improvements that can be made are highlighted, along with the limitations encountered in this study.

5.1. Main Conclusions

All of the objectives of this project were accomplished. It was possible to develop CNN models capable of correctly identifying various LULCs by DGT parameters. While the four architectures met the objectives, some performed better than others. The architecture designed for 10-band imagery analysis was the one that produced the best overall results across various performance metrics, as detailed above. However, it is also more computationally intensive and time-consuming in the training process, which raises questions about whether it should be preferred over the 3-band RGB custom model also defined in this study.

This 3-input channel model offers faster training times despite requiring more epochs to train (26 instead of 22 for the 10-band model). Each epoch on average is, however, quite quicker to train, requiring 1168.59 seconds when compared to 3311.15 seconds for the 10-band model. The difference between the two is minimal in terms of performance.

The transfer learning-based models both show worse performance when compared to the custom-designed models, achieving approximately 90% validation accuracy compared to around 97% and significantly higher loss (around 0.3 when in comparison with 0.07).

Additionally, this dissertation provides information on what concerns specific classes that are more troublesome in the classification, such as 'Annual Spring/Summer Cultures,' 'Maritime Pine,' 'Other Softwoods,' and 'Bushes'. These are the classes that consistently achieved lower classification results, probably because of their visual similarities or even due to the season characteristics that they show, given that the data analyzed is from the end of the summer. The difference in accuracy registered between the custom models and the transfer learning-based models may be attributed to the relatively small image size, only 33x33x3, as mentioned. The two networks used in the transfer learning process,

MobileNetV3 Large and EfficientNetB0, each expect ideal data with sizes of 224x224x3 and 256x224x3, respectively. To further improve accuracy and preserve more information, increasing the number of labeled pixels would be beneficial. The expansion of a larger number of training points would allow for the inclusion of more image patches and even enable pixel-level classification, which would make possible image segmentation. Other attempts were conducted with different image sizes with the use of resizing, but the results were not much different despite the increase in training time. With knowledge, the option was to not apply to resize, as it would only increase the time it would take to do each stage.

In what concerns the questions asked in Chapter 1, they will now be addressed explicitly:

(1) How can the accuracy of the LULC classification be improved while preserving the maximum amount of information?

    In an attempt to maintain the maximum amount of information, the initial approach was to do pixel-level classification. However, due to the number of labeled pixels being very low in the scale of the data, this wasn't possible. Instead, what was used were patches of the original image, which ended up being useful but a quite smaller dataset. With this new data, despite the stated smaller sample size, it was possible to achieve good model performance.

(2) How does the choice of CNN architecture affect the performance of terrain classification?

    The different CNN models achieve different performances, as would be expected. The first difference can be established by saying that the transfer learning models present lower performance when compared to the custom-defined models. Other than this, it is shown by the results that the model with more input channels generally outperforms the 3-band model.

(3) Can transfer learning be used to improve CNN's performance on MSI data for terrain classification? What are the most effective methods for fine-tuning pre-trained models, if applicable?

    From what was observed here, the use of transfer learning-based models was not capable of improving the classification performance when compared to custom-defined models, as previously shown. In regards to the aspect related to the fine-tuning of the parameters in this case, it was done with the changes in the learning rate and the change in the number of pre-trained layers that would or would not be trained with new data.

(4) How do different pre-processing strategies, like normalizing and reducing the number of dimensions, affect the performance of CNNs when using MSI data to classify terrain?

    Regarding the reduction of dimensions or bands, this resulted in a slight decrease in performance metrics. This decrease in the metrics is almost irrelevant

in the case of the custom model, which is the best to conduct this analysis given that the only difference is indeed the number of channels. In terms of data normalization, a comparison between two methods was made: one ranging from 0 to 255, used in transfer learning base models, and the other ranging from 0 to 1. While the second, used in custom models, performed better, it must be said that differences in performance aren't probably related only to normalization; other factors are also at play as the structures of models are also different with different parameters inside of them.

5.2. Limitations and Future Work

As stated previously, the designed architectures achieved success within the current context. However, the analysis can be enhanced. Currently, the analysis is restricted to ground patches of 33x33 pixels. Given the spatial resolution of 10m per pixel, this means that these patches cover an area of 108900 m^2. As this is a quite large area, the classification can certainly be improved.

One way for this work to be improved is if it were possible to change from patch classification to pixel-level classification. This would, however, involve the labeling of each single label before training, validation, and testing the models. Allowing for a finer analysis and making it possible to capture the details, in contrast to what is currently being used, which only has the central pixel with a ground truth label and relies on smaller parts of the image. This would also allow for less information loss and make it possible to improve the classification of smaller things in border areas.

For this change to be possible, two things would have to be done as stated; the second would depend on the first:

- Creating a data set where every single pixel of the images is labeled Maybe with the use of clustering algorithms to assign labels to individual pixels. An attempt was made at this during this thesis; however, as it didn't achieve good results and there wasn't enough time to invest here, the idea was just changed. This step would be fundamental to the next phase of implementation.
- Training new CNNs for pixel analysis would make it possible to even do image segmentation. These new models would make it possible to use all of the available information, taking full advantage of what was provided.

These new improvements are thought to be improvements in the right direction. Enhancing the classification of the images. However, it is important to make it clear that they present considerable challenges. The classification of every single pixel in the images is a process that may be very time-consuming and certainly computationally intensive. And also, depending on the algorithms chosen to do this work,

As an overview, despite the work done here achieving good results in proving that CNNs are a good alternative to the classifications of the images, the DGT, and their classifiers, which achieve accuracy values of around 70%, there is still large room for

improvement in this work. The proposed paths could lead to promising improvements in this context.

References

[1] T. Tian, C. Li, J. Xu, and J. Ma, "Urban area detection in very high resolution remote sensing images using deep convolutional neural networks," *Sensors (Switzerland)*, vol. 18, 3 2018.

[2] P. Griffiths, C. Nendel, and P. Hostert, "Intra-annual reflectance composites from sentinel-2 and landsat for national-scale crop and land cover mapping," *Remote Sensing of Environment*, vol. 220, pp. 135–151, 2019.

[3] C. Qiu, L. Mou, M. Schmitt, and X. X. Zhu, "Local climate zone-based urban land cover classification from multi-seasonal sentinel-2 images with a recurrent residual network," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 154, pp. 151–162, 8 2019.

[4] Y. La, H. Bagan, and Y. Yamagata, "Urban land cover mapping under the local climate zone scheme using sentinel-2 and palsar-2 data," *Urban Climate*, vol. 33, p. 100661, 9 2020.

[5] A. Ozdarici-Ok, A. O. Ok, and K. Schindler, "Mapping of agricultural crops from single high-resolution multispectral images—data-driven smoothing vs. parcel-based smoothing," *Remote Sensing 2015, Vol. 7, Pages 5611-5638*, vol. 7, pp. 5611–5638, 5 2015.

[6] C. Zhang and J. M. Kovacs, "The application of small unmanned aerial systems for precision agriculture: A review," *Precision Agriculture*, vol. 13, pp. 693–712, 12 2012.

[7] L. Moser, G. Ramminger, M. Probeck, C. Rieke, B. Mack, C. Storch, C. Sommer, C. Sandow, R. Richter, M. Sindram, A. Homolka, H. Ott, M. Ickerott, and A. Relin, "Crop mapping for a future copernicus agricultural service," *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pp. 3848–3851, 7 2018.

[8] P. Defourny, S. Bontemps, N. Bellemans, C. Cara, G. Dedieu, E. Guzzonato, O. Hagolle, J. Inglada, L. Nicola, T. Rabaute, M. Savinaud, C. Udroiu, S. Valero, A. Bégué, J.-F. Dejoux, A. E. Harti, J. Ezzahar, N. Kussul, K. Labbassi, V. Lebourgeois, Z. Miao, T. Newby, A. Nyamugama, N. Salh, A. Shelestov, V. Simonneaux, P. S. Traore, S. S. Traore, and B. Koetz, "Near real-time agriculture monitoring at national scale at parcel resolution: Performance assessment of the sen2-agri automated system in various cropping systems around the world," *Remote Sensing of Environment*, vol. 221, pp. 551–568, 2019.

[9] E. Roteta, A. Bastarrika, M. Padilla, T. Storm, and E. Chuvieco, "Development of a sentinel-2 burned area algorithm: Generation of a small fire database for sub-saharan africa," *Remote Sensing of Environment*, vol. 222, pp. 1–17, 3 2019.

[10] S. Hislop, S. Jones, M. Soto-Berelov, A. Skidmore, A. Haywood, and T. H. Nguyen, "Using landsat spectral indices in time-series to assess wildfire disturbance and recovery," *Remote Sensing*, vol. 10, 2018.

[11] P. Leinenkugel, R. Deck, J. Huth, M. Ottinger, and B. Mack, "The potential of open geodata for automated large-scale land use and land cover classification," *Remote Sensing*, vol. 11, 2019.

[12] W. S. McCulloch and W. H. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, vol. 5, pp. 115–133, 1943.

[13] A. Turing, "Intelligent machinery (1948)," *The Essential Turing*, 9 2004.

[14] A. M. Turing, "Computing machinery and intelligence," *Mind*, vol. 59, pp. 433–460, 1950.

[15] J. Macqueen, "Some methods for classification and analysis of multivariate observations," *Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967.

[16] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, pp. 193–202, 4 1980.

[17] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Computation*, vol. 1, pp. 541–551, 3 1989.

[18] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, 11 1998.

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, pp. 211–252, 2015.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, 2012.

[21] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS Quarterly: Management Information Systems*, vol. 28, pp. 75–105, 2004.

[22] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of Management Information Systems*, vol. 24, pp. 45–77, 12 2007.

[23] Y. Zhou, Y. Xu, P. Sekula, and L. Ding, "Machine learning in construction: From shallow to deep learning," *Developments in the Built Environment*, vol. 6, 3 2021.

[24] C. Cortes, V. Vapnik, and L. Saitta, "Support-vector networks," *Machine Learning 1995 20:3*, vol. 20, pp. 273–297, 9 1995.

[25] P. Zhang, "Satellite image classification based on convolutional neural network," *2021 3rd International Academic Exchange Conference on Science and Technology Innovation, IAECST 2021*, pp. 754–757, 2021.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[27] K. Kolodiazhnyi, "Hands-on machine learning with c++." `https://www.oreilly.com/library/view/hands-on-machine-learning/9781789955330/d08009e2-d7ab-49b3-a2ad-9f8c360d720c.xhtml`, 2023.

[28] S. Panigrahi, "What are hyper-spectral images?." Medium, Apr. 2021. accessed Jan. 17, 2023.

[29] B. Kitchenham, "Procedures for performing systematic reviews," technical report tr/se-0401, Keele University, Department of Computer Science, Keele University, UK, 2004.

[30] Z. Kong and H. T. Yang, "Hyperspectral image classification method based on machine learning," *2021 IEEE International Conference on Computer Science, Electronic Information Engineering and Intelligent Control Technology, CEI 2021*, pp. 392–395, 9 2021.

[31] X. Liu, H. Wang, J. Liu, S. Sun, and M. Fu, "Hsi classification based on multimodal cnn and shadow enhance by dsr spatial-spectral fusion," *Canadian Journal of Remote Sensing*, vol. 47, pp. 773–789, 2021.

[32] M. Verma, N. Gupta, B. Tolani, and R. Kaushal, "Explainable custom cnn architecture for land use classification using satellite images," *Proceedings of the IEEE International Conference Image Information Processing*, vol. 2021-November, pp. 304–309, 2021.

[33] U. Singh, K. Saurabh, N. Trehan, R. Vyas, and O. P. Vyas, "Terrain classification using transfer learning on hyperspectral images: A comparative study," *2022 IEEE 19th India Council International Conference (INDICON)*, pp. 1–6, 11 2022.

[34] S. K. Roy, G. Krishna, S. R. Dubey, and B. B. Chaudhuri, "Hybridsn: Exploring 3-d-2-d cnn feature hierarchy for hyperspectral image classification," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, pp. 277–281, 3 2020.

[35] Y.-L. Chang, T.-H. Tan, W.-H. Lee, L. Chang, Y.-N. Chen, K.-C. Fan, and M. Alkhaleefah, "Consolidated convolutional neural network for hyperspectral image classification," *Remote Sensing*, vol. 14, 2022.

[36] K. Hara, H. Kataoka, and Y. Satoh, "Learning spatio-temporal features with 3d residual networks for action recognition," *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017*, vol. 2018-January, pp. 3154–3160, 8 2017.

[37] G. Sumbul, M. Charfuelan, B. Demir, and V. Markl, "Bigearthnet: A large-scale benchmark archive for remote sensing image understanding," in *IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium*, IEEE, jul 2019.

[38] K. Ali and B. A. Johnson, "Land-use and land-cover classification in semi-arid areas from medium-resolution remote-sensing imagery: A deep learning approach," *Sensors*, vol. 22, 2022.

[39] P. M, T. K, and C. Ganapathi, "Evaluation of land use/land cover classification based on different bands of sentinel-2 satellite imagery using neural networks," *International Journal of Advanced Computer Science and Applications*, vol. 13, 01 2022.

[40] N. Zaabar, S. Niculescu, and M. K. Mihoubi, "Assessment of combining convolutional neural networks and object based image analysis to land cover classification using sentinel 2 satellite imagery (tenes region, algeria)," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLIII-B3-2021, pp. 383–389, 2021.

[41] European Space Agency (ESA), "Sentinel-2 multispectral instrument (msi) - revisit and coverage," Accessed 2023. Accessed on 14/08/2023.

[42] European Space Agency (ESA), "Sentinel-2 multispectral instrument (msi) - level-2a product types," Accessed 2023. Accessed on 13/08/2023.

[43] L. Baetens, C. Desjardins, and O. Hagolle, "Validation of copernicus sentinel-2 cloud masks obtained from maja, sen2cor, and fmask processors using reference cloud masks generated with a supervised active learning procedure," *Remote Sensing*, vol. 11, no. 4, p. 433, 2019.

[44] J. W. Rouse, R. H. Haas, D. W. Deering, and J. A. Schell, "Monitoring the vernal advancement and retrogradation (green wave effect) of natural vegetation," Tech. Rep. RSC 1978-4, Remote Sensing Center, Texas A&M Univ., College Station, 1974.

[45] S. Hislop, S. Jones, M. Soto-Berelov, A. Skidmore, A. Haywood, and T. H. Nguyen, "Using landsat spectral indices in time-series to assess wildfire disturbance and recovery," *Remote Sensing*, vol. 10, no. 3, p. 460, 2018.

[46] S. K. McFeeters, "The use of the normalized difference water index (ndwi) in the delineation of open water features," *International journal of remote sensing*, vol. 17, no. 7, pp. 1425–1432, 1996.

[47] Y. Zha, J. Gao, and S. Ni, "Use of normalized difference built-up index in automatically mapping urban areas from tm imagery," *International journal of remote sensing*, vol. 24, no. 3, pp. 583–594, 2003.

[48] E. Roteta, A. Bastarrika, M. Padilla, T. Storm, and E. Chuvieco, "Development of a sentinel-2 burned area algorithm: Generation of a small fire database for sub-saharan africa," *Remote sensing of environment*, vol. 222, pp. 1–17, 2019.

[49] Direção-Geral do Território, "Especificações técnicas da carta de uso e ocupação do solo (cos) de portugal continental para 1995, 2007, 2010, 2015 e 2018," relatório técnico, Direção-Geral do Território, 2022. Coordenação: Mário Caetano e Filipe Marcelino; Equipa técnica e de investigação e desenvolvimento: Cristina Roxo, David Francisco, Dénis Andrade, Francisco Moreira, Flávio Oliveira, Giselda Monteiro, Hugo Costa, Inês Machado, Joana Laurentino, José Tomé, Pedro Benevides, Rogério Madeira, Rui Carvalho.

[50] A. Howard, M. Sandler, G. Chu, L. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, Q. V. Le, and H. Adam, "Searching for mobilenetv3," *CoRR*, vol. abs/1905.02244, 2019.

[51] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," *CoRR*, vol. abs/1905.11946, 2019.

MDPI

Article

Land cover automatic classification using deep learning techniques applied to satellite imagery

Sérgio Santos [1], Tomás Brandão [1,2] and Luís Nunes [1,2]*

[1] Iscte - Instituto Superior Universitário
[2] ISTAR_Iscte
* Correspondence: luis.nunes@e-mail.com;
† Current address: Affiliation 3.
‡ These authors contributed equally to this work.

Abstract: In an era of population growth and rapid urbanization, sustainable urban development is crucial. Machine learning (ML) emerges as a key player, facilitating the swift processing of remote sensing images. This paper centers on leveraging Convolutional Neural Networks (CNNs) for multispectral image handling. The primary goal is to evaluate the performance metrics and computational complexity of a CNN-based land cover classification approach, comparing results with the Direção Geral do Território (DGT) architectures. Initial steps involve comprehending the provided data, preprocessing it, and defining the architecture. Results indicate CNNs as a promising alternative for land cover classification, though challenges, notably the scarcity of labeled data, underscore opportunities for improvement. Despite promising outcomes, the work highlights areas for enhancement, particularly in data preprocessing. In summary, the investigation into CNNs for land cover classification yields positive results with room for expected improvement, especially in data preprocessing.

Keywords: Multispectral imaging, Convolutional Neural Networks, Machine Learning, Land Use Land Cover Classification, Sentinel-2.

1. Introduction

Numerous aspects of land governance and sustainable development rely on Land Use and Land Cover mapping through the classification of Remote sensing images. This classification allows government entities and companies to make informed decisions regarding its use.

For example, in the context of urban planning , agriculture , fire prevention , or even mineral exploration, the use of LULC mappings allows for better planning. In addition, it aids in comprehending the planet as a system; despite being frequently used interchangeably, land cover and land use are not synonymous. Land cover refers to the types of features present on the earth's surface, including, among others, trees, rocks, lakes, and roads. However, land use refers to the specific activity or function being developed on that piece of land, such as urbanization or residential development. Comparatively, when referring to a neighborhood, the term land cover would refer to the road, roofs, grass, and trees, whereas the term land use would refer to residential use. The significance and value of these High resolution remote sensing images have grown as their technology, availability, and quality improved.

Moreover, because of the opportunities they afford, they provide access to vast quantities of data that span vast areas of land in detail and over extensive periods. Therefore, most land studies involve analyzing satellite or aircraft-mounted sensor photographs. Nevertheless, the size and variety of data types in these images made them challenging to process, further complicating the situation. A single pixel may hold spatial, spectral, and geometric data, among other things.

Citation: Lastname, F.; Lastname, F.; Lastname, F. Title. *Journal Not Specified* **2023**, *1*, 0. https://doi.org/

Received:
Revised:
Accepted:
Published: