Politecnico Di Torino

Master Of Science Program In Civil Engineering LM-23
Collegio di Ingegneria Civile

Corridor Mapping Processing Using the Machine Learning Approach

Supervisors:

- Prof. Paolo Dabove(DIATI,Polito)
- Eng. Olivotto Luca (DigiSky SRL)

Candidate:

- Muhammad Daud
  s287701

Acknowledgments

I want to take this opportunity to express my heartfelt gratitude to my academic supervisor, Prof. Paolo Dabove, and my external company supervisor, Eng. Olivotto Luca, for their invaluable guidance and support throughout my master's research thesis. Prof. Paolo Dabove, with his expertise in Remote sensing, has imparted me with the necessary skills and knowledge to complete this challenging project successfully. Eng. Luca and DIGISKY SRL provided me with the datasets and technical inputs that were fundamental to the success of my work. I am deeply grateful to both for their trust in me and for taking a chance on this new and exciting topic.

I also want to express my heartfelt gratitude to my elder sister, Arooj Niaz, for her unwavering support and encouragement throughout my academic journey. I could not have completed this journey without her unwavering support and motivation. Once again, I would like to extend my heartfelt appreciation to all those who have contributed to my success.

ستاروں سے آگے جہاں اور بھی ہیں

Table of Contents

List of Illustrations

List of Tables

List of Glossary of Acronyms

AI: Artificial Intelligence

ML: Machine Learning

DL: Deep Learning

CNN: Convolutional Neural Network

RF: Random Forest algorithm

SVM: Support Vector Machine

U NET: U-shaped convolutional neural network

CNN-RNN: Convolutional Neural Network - Recurrent Neural Network

ResNet: Residual Network

OBIA: Object-Based Image Analysis

PBIA: Pixel-Based Image Analysis

GIS: Geographic Information System

RS: Remote Sensing

RGB: Red, Green, Blue

DEM: Digital Elevation Model

MSI: Multi-Spectral Image

HYS: Hyperspectral Image

LiDAR: Light Detection and Ranging

InSAR: Interferometric Synthetic Aperture Radar

UAV: Unmanned Aerial Vehicle

NDVI: Normalized Difference Vegetation Index

GCP: Ground Control Points

LULC: Land Use and Land Cover

ROI: Region of Interest

TP: True Positive

TN: True Negative

FP: False Positive

FN: False Negative

F1 Score: A Measure of Test's Accuracy That Considers Precision and Recall.

Abstract.

The study investigates the use of machine learning in remote sensing to identify and map linear features such as urban features, roads, pipelines, and utilities in civil engineering. Remote sensing involves using sensors to gather data about the Earth's surface and atmosphere from a distance. Machine learning is a powerful tool that can be used to analyze and interpret this data to extract meaningful insights and make predictions. The availability of large amounts of data from remote sensing technologies has greatly increased in recent years. It has become increasingly more work to analyze and interpret this data manually. Machine learning solves this problem by automating the analysis and interpretation of remote sensing data.

Using machine learning in remote sensing can provide numerous benefits, including increased efficiency and accuracy in data analysis. In this study's context, the primary dataset's accuracy, which only contained RGB values, was 83%. The accuracy was improved to 89.9% when an integrated dataset containing RGB, and elevation information was used. The study also compared pixel-based and object-based classification using a random forest algorithm and found that object-based classification led to slightly improved accuracy. Furthermore, the accuracy was improved from 89.9% to 92.08% when using a deep learning convolutional neural network (CNN), even for a tenfold larger area. The training time for the CNN algorithm model was 6x longer than the traditional machine learning model. However, implementing the trained model over large areas took only minutes rather than hours and enabled the extraction of single types of features without any redundant data. Similar results (91.11%) were obtained in a second example applied to a road corridor.

The findings of this study can have practical applications, such as the analysis of structures & infrastructure safety and digital twins. Machine learning can analyze data from remote sensing over time to detect changes and trends, which can be useful for understanding environmental impacts and identifying potential conflicts. Additionally, the results of this study can inform the design and planning of urban corridors with data credibility in mind by helping to identify the potential hazard and possible solutions.

1 Chapter: Introduction and Background

1.1 Introduction:

Remote sensing involves using sensors to gather data about the Earth's surface and atmosphere from a distance. Machine learning is a powerful tool that can be used to analyze and interpret data to extract meaningful insights and make predictions.

Using machine learning in remote sensing can provide numerous benefits, including increased efficiency and accuracy in data analysis, versatility in various applications, and the ability to scale up to handle large datasets easily. These advantages make machine learning a powerful tool for understanding and managing the environment and enabling more informed decision-making about the conservation and management of natural and urban environments.

Some examples of how machine learning is used in remote sensing include:

- *Land cover classification:* Machine learning algorithms can be trained to recognize different types of land cover (e.g., forests, grasslands, urban areas) based on features extracted from remote sensing data. This can be used to map the distribution of different land cover types, monitor land use change, and assess the health of ecosystems.
- *Object detection and tracking:* Machine learning algorithms can identify and track objects in remote sensing data, such as vehicles, ships, or aircraft. This can be useful for various applications, including transportation and logistics, defense, and environmental monitoring.
- *Image classification:* Machine learning can classify images based on their content, such as distinguishing between different vegetation types, identifying the presence of water, or detecting changes over time. This can be useful for various applications, including agriculture, natural resource management, and disaster response.
- *Corridor mapping:* The process of identifying and mapping the boundaries of corridors, defined as linear areas connecting two or larger areas, such as forests or wetlands. Machine learning can be used to automate this process by training algorithms to recognize the characteristics of corridors in remote sensing data.

1.2 Background:

In civil engineering, corridor mapping is often used to identify and map linear features such as roads, pipelines, and utilities. Some common types of corridor mapping in civil engineering include:

- Pipeline corridor mapping involves identifying and mapping the boundaries of pipelines and the surrounding rights-of-way. Pipeline corridor mapping can be used to understand the impacts of pipeline construction on the environment and to identify potential conflicts with other land uses or infrastructure.

- Utility corridor mapping: This involves identifying and mapping the boundaries of utility corridors, which contain infrastructures such as power lines, water lines, and communication cables. Utility corridor mapping can be used to understand the impacts of utility construction on the environment and to identify potential conflicts with other land uses or infrastructure.

- Road corridor mapping involves identifying and mapping the boundaries of roads and the surrounding rights-of-way. Road corridor mapping can be used to understand the impacts of road construction on the environment and to identify potential conflicts with other land uses or infrastructure.

- Urban corridor mapping involves identifying and mapping corridors within urban environments that contain infrastructures such as roads, utilities, and other linear features. Urban corridors may also include green spaces, parks, and other areas providing wildlife habitat or connectivity.

Urban corridors are typically characterized by high land use and development levels and can be a major focus of infrastructure planning and investment in urban areas. Urban corridors may be important for transportation, economic development, and the quality of life of urban residents. In civil engineering, the planning and design of urban corridors may involve a wide range of considerations, including traffic flow, pedestrian and bicycle access, environmental impacts, and land use patterns. Civil engineers may work with other professionals such as planners, architects, and environmental scientists to understand urban communities' needs and priorities and develop infrastructure and land use plans that are sustainable and equitable.

Manual mapping of urban corridors is a time-consuming and tedious process. However, machine learning can help automate this process, making it more efficient and accurate. This study examined the challenges and opportunities of using machine learning for urban mapping corridors to understand how it can be applied effectively in this context.

One of the challenges of using urban corridor mapping in remote sensing is the complexity of the data, as urban environments often contain a wide range of features with different characteristics and properties. For example, buildings may vary in size, shape, and height, and

roads may vary in width, surface type, and traffic volume. Also, the radiometric contents of different urban features can be similar, yet there represent different meanings on the ground. This study investigated two examples: an urban corridor and a road corridor. The urban corridor investigation aimed to recognize the complexity and dependency of different features in an urban environment. The results of the first investigation were then used to carry out the second study in a road environment with less complexity and more defined features.

1.3 Research Objectives

As stated above, corridor mapping deals with a larger area, which means an excess of information at hand with time; it is hard to handle the data into information. This is where machine learning can be useful, as it allows for the automated analysis and interpretation of data at scale. Machine learning algorithms can be trained to recognize patterns and trends in data that may not be apparent to the human eye and can help to extract meaningful insights and predictions from large datasets.

Machine learning can significantly reduce the time and effort required to analyze and interpret corridor mapping, allowing for more efficient and accurate decision-making. Machine learning can also help to improve the accuracy and precision of corridor mapping, as algorithms can be trained to recognize a complex pattern in data that may not be visible to the human eye. The following are the research objectives that were studied in the study:

1. **Objective 1:** To evaluate the effectiveness of current state-of-the-art machine learning approaches in corridor mapping using remote sensing data.
2. **Objective 2:** To assess the impact of adding additional data bands on the classification of urban features in corridor mapping.
3. **Objective 3:** To compare the performance of pixel-based and object-based classification methods in corridor mapping.
4. **Objective 4:** Comparison of Machine learning (ML) vs. Deep learning (DL) in real-life corridor mapping applications.
5. **Objective 5:** To evaluate the accuracy and computational time of different ML and DL algorithms for corridor mapping.

1.4 Research Approach:

The following is the approach implemented to get the desired outcome:

1. *Define the research question:* The first step in any research study is clearly defining the research question or problem the study aims to address. In this case, the research question is:

"What are the most effective approaches for using machine learning to recognize environment features in remote sensing data?"

2. *Collect and prepare the data:* The next step is collecting and organizing the study's data. This involves acquiring remote sensing data, such as ortho mosaic and DEM data, and processing the data to extract features used as inputs to machine learning algorithms.
3. *Train and evaluate machine learning models:* The next step is to train machine learning algorithms to recognize patterns and trends in the data. This involves using techniques such as supervised learning, in which algorithms are trained to classify data based on known labels, or unsupervised learning, in which algorithms are trained to identify patterns in data without prior labels. The performance of the algorithms can then be evaluated using metrics such as accuracy, precision, and recall.
4. *Analyze and interpret the results:* The next step is to analyze and interpret the results once the machine learning models have been trained and evaluated. This involves using techniques such as feature importance to understand which features in the data are most important for predicting the outcome of interest and visualizing the results to understand patterns and trends in the data.
5. *Communicate the results:* The final step is communicating the study results, which involves writing a research paper or a product or presenting the findings to a wider audience. It is important to communicate the research questions, methods, results, and implications of the study to share the findings with others and to contribute to the broader field of knowledge.

1.5 Thesis Outline:

The thesis outline provides an overview of the structure and content of the research study. It includes chapters on introduction, literature review, spatial domain, methodology, implementation, accuracy comparisons, and prospects for future research.

1.5.1 Chapter I. Introduction and Background

The first chapter of the thesis serves as an introduction to the research and provides a brief overview of the background information. The chapter also outlines the research objectives and approach that will be taken in the subsequent chapters. Finally, the chapter provides an outline of the overall thesis structure.

1.5.2 Chapter II. Literature Review

The second chapter is a literature review that summarizes the existing research and studies conducted on using machines and deep learning in remote sensing. This chapter discusses the relevant theories, methodologies, and applications developed and applied to the field.

1.5.3 Chapter III. Spatial Domain

Chapter three focuses on the study area and provides a detailed description of the data used in the study. The chapter covers the different types of remote sensing data and the various sources from which the data is obtained.

1.5.4 Chapter IV. Methodology

Chapter four describes the methodology that will be used in the study. The chapter outlines the definitions of key terms, the system specifications, and the software platforms used. The chapter also provides a detailed explanation of the methodology employed in the research.

1.5.5 Chapter V. Implementation

Chapter five describes the methodology's implementation outlined in the previous chapter. The chapter covers unsupervised classification, supervised classification, pixel-based supervised classification, pixel-based classification on primary and integrated datasets, object-based image classification, real-life application, and implementation of DL using convolution neural networks(U-Net).

1.5.6 Chapter VI. Accuracy Comparisons

Chapter six focuses on the accuracy comparisons of the different methodologies used in the study. This chapter discusses the accuracy assessment tools used, the comparison of primary and integrated datasets, pixel-based vs. object-based image classification, and one feature extraction for corridor analysis.

1.5.7 Chapter VII. Prospects

The final chapter provides a conclusion to the thesis and discusses the prospects for future research. The chapter summarizes the study's key findings and offers suggestions for future research and machine and deep learning application in remote sensing.

2 Chapter: Literature Review

In recent years, researchers have explored machine learning methods in various remote sensing applications, including land use classification, crop monitoring, and environmental monitoring. The combination of remote sensing and machine learning techniques has the potential to revolutionize our understanding of the earth's surface, providing insights into a wide range of scientific and societal challenges. This literature review will explore the field's current state and identify key areas for future research.

Lu, Zheng, and Yuan (Lu et al., 2017) proposed an unsupervised representation learning method to investigate deconvolution networks for remote sensing scene classification. She achieved experimental results that outperformed most state-of-the-art techniques.

In a study (Myint et al., 2011) using high-resolution Quick Bird image data, a comparison was made between per-pixel and object-based classifiers in identifying urban land cover classes in Phoenix, Arizona. The study found that the object-based classifier achieved significantly higher overall accuracy than the per-pixel maximum likelihood classifier.

Belgiu and Drăguţ (Belgiu & Drăgu, 2016) reviewed the applications and future directions of the random forest (RF) classifier in Remote Sensing. They found that the RF classifier is popular in remote sensing due to its accuracy and ability to handle high data dimensionality and multicollinearity. Noi and Kappas (Thanh Noi & Kappas, 2017) compared the performances of Random Forest (RF), k-Nearest Neighbor (kNN), and Support Vector Machine (SVM) classifiers for land use/cover classification using Sentinel-2 image data. They found that all three classifiers showed high overall accuracy ranging from 90% to 95%. Swapan &Talukdar (Talukdar, 2020) reviewed six machine-learning algorithms for land-use/land-cover (LULC) mapping using earth observations and found that the random forest (RF) algorithm had the highest accuracy level among the examined classifiers.

Prakash, Manconi, and Loew (Prakash et al., 2020) compared the performance of deep learning models with traditional machine learning models for mapping landslides from Earth observation (EO) data. Lei Ma & Yu Liu (Ma et al., 2019) conducted a meta-analysis and review of over 200 publications on deep learning (DL) algorithms in remote-sensing image analysis, introducing the major DL concepts in remote Sensing and reviewing various remote-sensing image analysis tasks. Xin & Zhang (X. Zhang et al., 2020) comprehensively reviewed land cover classification and object detection using high-resolution remote sensing imagery and found that deep learning models outperform traditional pixel-based methods, particularly for vegetation categories.

U-Net models have proven effective and efficient for various tasks related to urban analysis from satellite and aerial imagery. Francini et al.(Francini et al. 2023) used a U-Net model to accurately classify and segment the built-up area and the vegetation cover in Matera (Italy) over a period of 20 years. They demonstrated how their method could support sustainable development by providing valuable information about urban and greening changes.

Kumar et al.(Chaudhary et al., 2022) enhanced the U-Net architecture by adding batch normalization, dropout, and residual connections to improve the performance and generalization of the model for road network segmentation. They achieved better results than several baseline models on two public datasets.

In conclusion, the literature review highlights the increasing use of machine learning and deep learning algorithms for remote sensing image analysis. Random Forest and U-Net models have emerged as better options for analyzing aerial imagery through machine learning techniques. These models have demonstrated high accuracy and efficiency in various remote sensing applications, surpassing traditional machine learning models and pixel-based methods. Random Forest's ability to handle high-dimensional data and U-Net's effectiveness in urban analysis and road network segmentation have shown their potential to revolutionize the understanding of the Earth's surface and support sustainable development processes.

3 Chapter: Spatial Domain

This chapter provides an overview of the two study areas selected for the research: the city of Turin and a road corridor from Napoli to Salerno in Italy. It also describes the two primary input rasters used for each case study: the orthomosaic and the Digital Elevation Model (DEM).

3.1 Study Area:

The research carried out two investigations to achieve its objectives: one focused on the urban environment and the other on a road corridor. For each analysis, an Italian study area was selected.

- The first study area was the city of Turin, Italy's densely populated area of 22.9 km^2 with similar radiometric content.
- The second case study was a road corridor from Napoli to Salerno, a length of 54.5 km, via the A3 motorway.

For each case study, two rasters were used: an orthomosaic with 25 pixels resolution and a Digital Elevation Model (DEM) with 50 cm resolution. Both rasters were obtained from DIGISKY SRL. Using these rasters, created two types of datasets, which are described in the following sections.

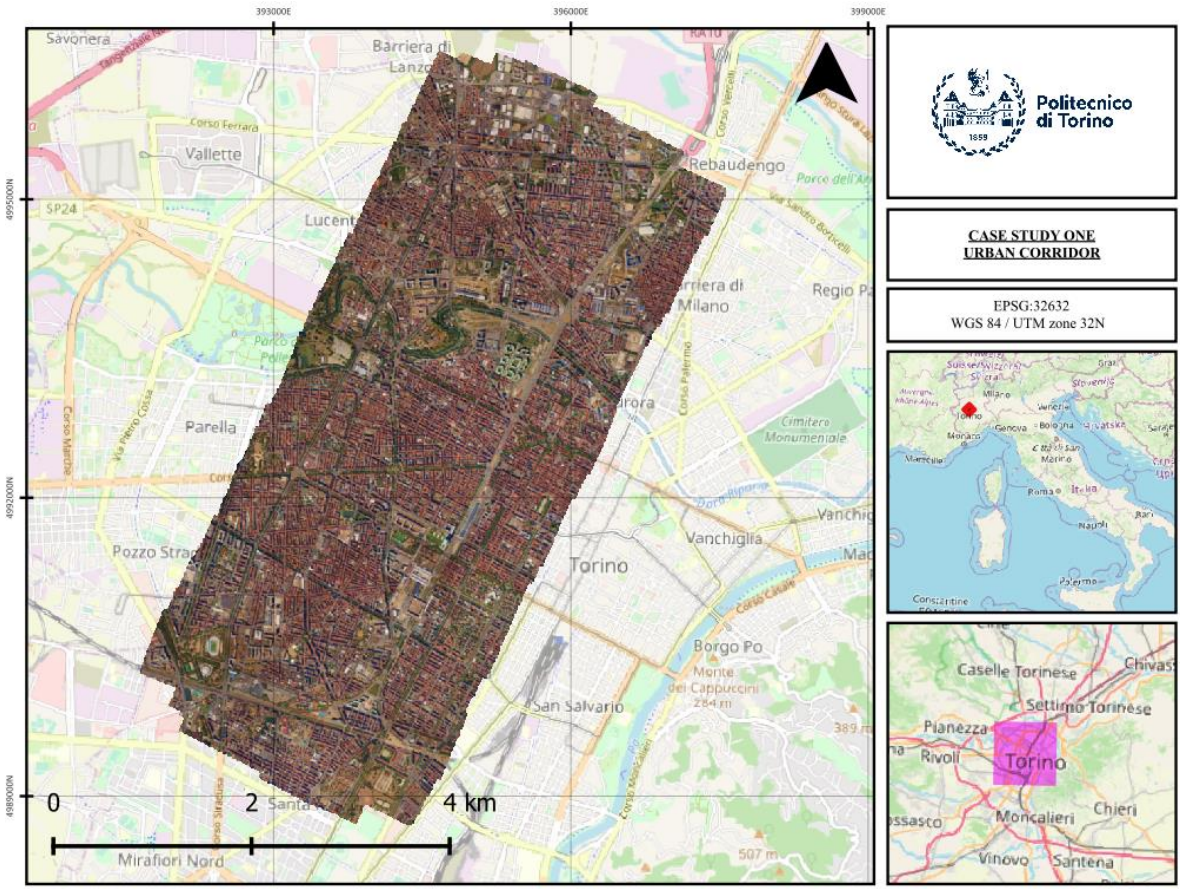


Figure 3.1 Case Study One Urban Corridor

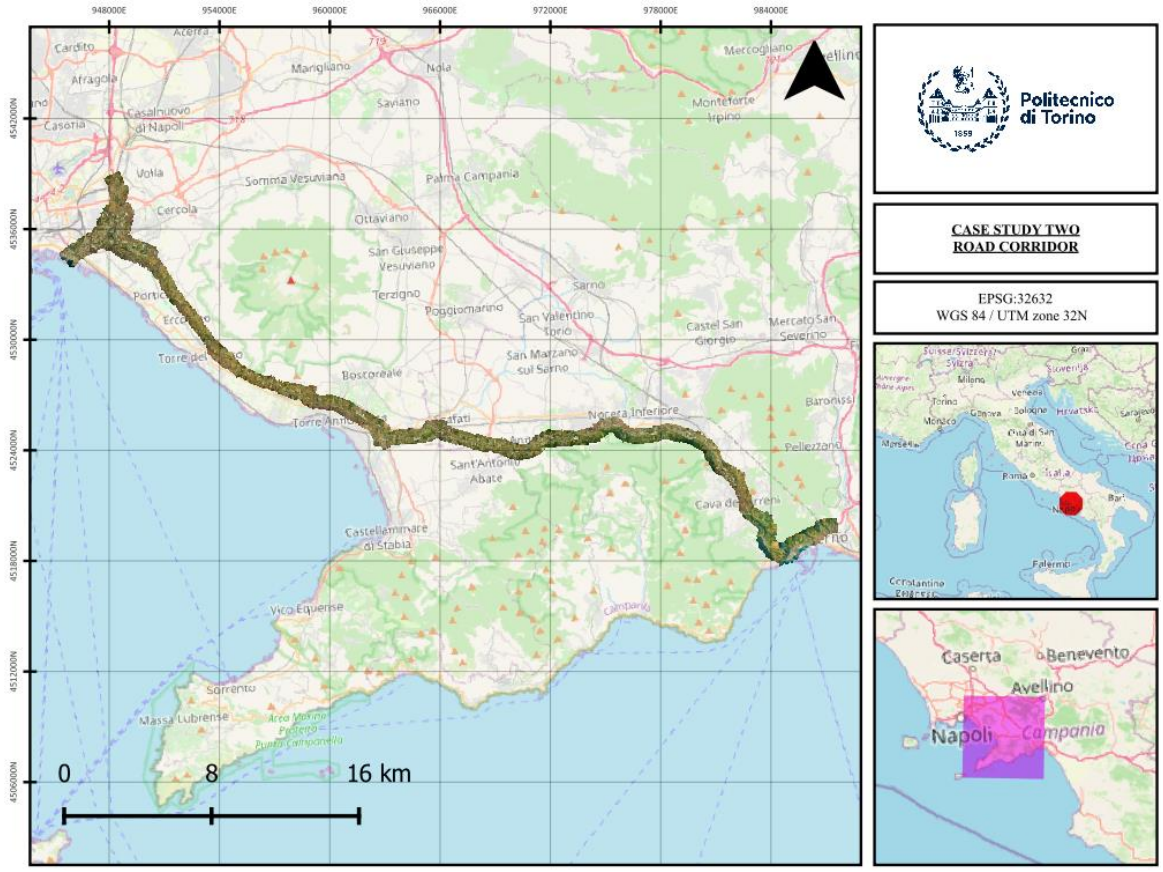


Figure 3.2 Case Study Two Road Corridor

3.2 Data Description:

The two primary input rasters are ortho mosaic and DEM. The source of both rasters is the same, using photogrammetry techniques via an aerial platform.

Orthomosaic and DEMs are two digital maps commonly used for understanding and mapping the Earth's surface. Orthomosaics, or orthorectified mosaics, are created by combining multiple aerial or satellite images(orthophotos) of the same area into a single, seamless orthophoto. These aerial photographs are corrected for distortion and can be used as base map layers in GIS and other mapping applications.

- Digital Elevation Models (DEMs) are digital representations of terrain elevation data that can be obtained through several methods, including aerial LiDAR, satellite imagery, photogrammetry, terrestrial LiDAR, stereographic techniques, digitized contour lines, and field surveys. The DEM is created using photogrammetry, so both raster sources are identical. To obtain a Digital Elevation Model (DEM) from photogrammetry, one needs to acquire high-resolution photographs of the area one wants to create the DEM for and use photogrammetry software to process the pictures.
- Orthophotos are typically larger in scale, higher in resolution, and more accurate than DEMs. However, DEMs are smaller in scale and may be lower in resolution due to the limitations of the sensors used to collect the data. Both maps are useful for different purposes and have different strengths and limitations.

This study created two datasets for each case study using two rasters: an orthophoto raster with 3 RGB bands and a DEM raster with a single band representing elevation.

1. The first dataset, the Primary Dataset1, consists only of the ortho mosaic, with three band values per pixel representing a feature.
2. The second dataset, Integrated Dataset 2, includes the ortho mosaic (3 bands) and the DEM as a fourth band. A single feature in this dataset is represented by four band values, including the elevation data.

The performance of the datasets can be evaluated by comparing them in terms of accuracy and computational time in both study areas using different techniques. Moreover, the literature suggests that integrated datasets can be useful in environments where height information is crucial for defining features.

Note: Due to limited computational capacity, the machine learning algorithm is only applied to 10% of the urban corridor studied in case 1, covering an area of 2 square kilometers. However, the results are expected to reflect the general area since the environment remains unchanged. For deep learning, the whole area is being processed.

Type	Description
Primary Dataset	Consists only of an orthomosaic with three band values per pixel representing a feature. Orthomosaics are created by combining multiple aerial or satellite images of the same area into a single, seamless orthophoto. They are typically larger in scale, higher in resolution, and more accurate than DEMs. This dataset is useful for mapping and visualizing features on the Earth's surface.
Integrated Dataset	Includes the orthomosaic and the DEM as a fourth band, so a single feature in this dataset is represented by four band values, including the elevation data. Digital Elevation Models (DEMs) are digital representations of terrain elevation data that can be obtained through several methods, including photogrammetry. This dataset is useful in environments where height information is important for defining features. However, it may be lower in resolution due to the limitations of the sensors used to collect the data.

Table 3.1 Dataset Description

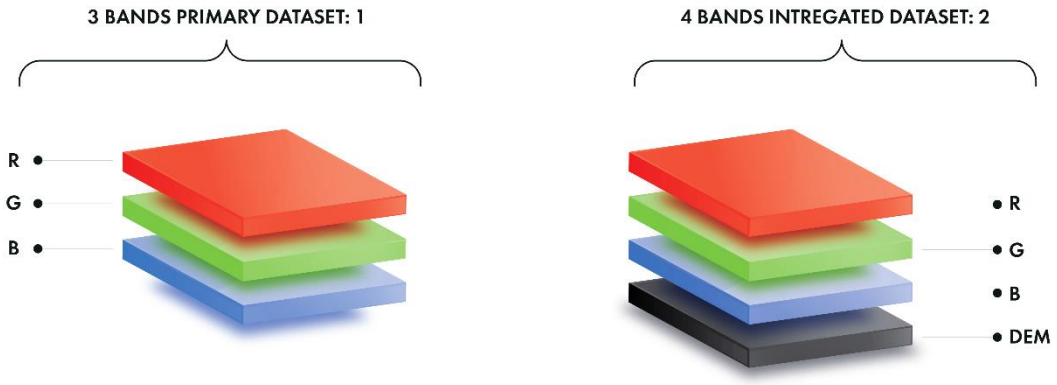


Figure 3.3 Datasets Comparison

4 Chapter: Methodology

The methodology chapter defines important terms and concepts related to artificial intelligence and machine learning. It also provides an overview of different types of machine learning algorithms commonly used in remote sensing applications. This information serves as the foundation for the research methodology employed in the study.

4.1 Definitions:

An artificial intelligence (AI) system is a process that can perform/pretends to perform tasks normally handled by humans, such as learning, problem-solving, and making decisions.

The concept of machine learning can be described as a subset of artificial intelligence (AI) in which computers can learn from data without explicitly programming them. In machine learning, a model is trained on a dataset and makes predictions or decisions based on that training. As the model is exposed to more data, it can improve its performance and become more accurate in its predictions.

There are several types of machine learning, including:

- Supervised learning: The model is trained on labeled data, including input and correct output. The model makes predictions based on this training data. Examples of supervised learning tasks include regression and classification.
- Unsupervised learning: In unsupervised learning, the machine has no structured data for training. The model recognizes the patterns in the provided data and comes up with classifications. Examples of unsupervised learning tasks include clustering and dimensionality reduction.
- Semi-supervised learning: Semi-supervised learning involves a combination of labeled and unlabeled data. It is often used when unlabeled data is available, but labeling it is too time-consuming or expensive.
- Reinforcement learning: In reinforcement learning, the model is trained to achieve a specific goal through trial and error. The model receives rewards for successful actions and punishments for unsuccessful ones, and it learns to make decisions based on these rewards and punishments. Reinforcement learning is used in applications such as self-driving cars and game-playing.

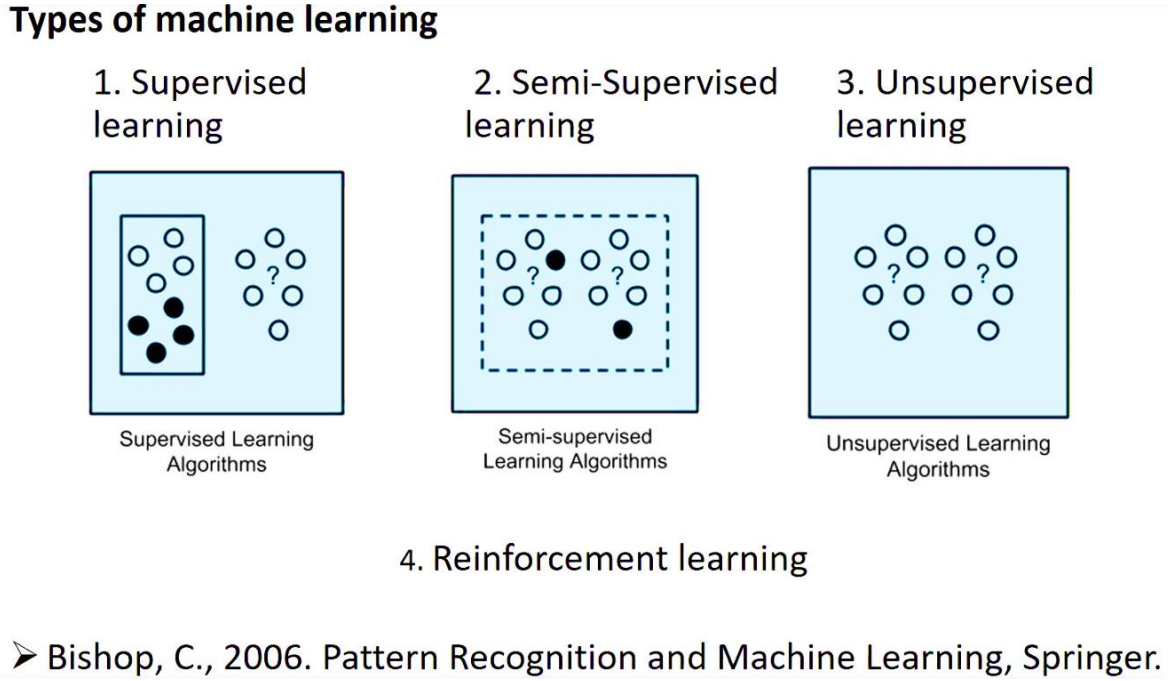


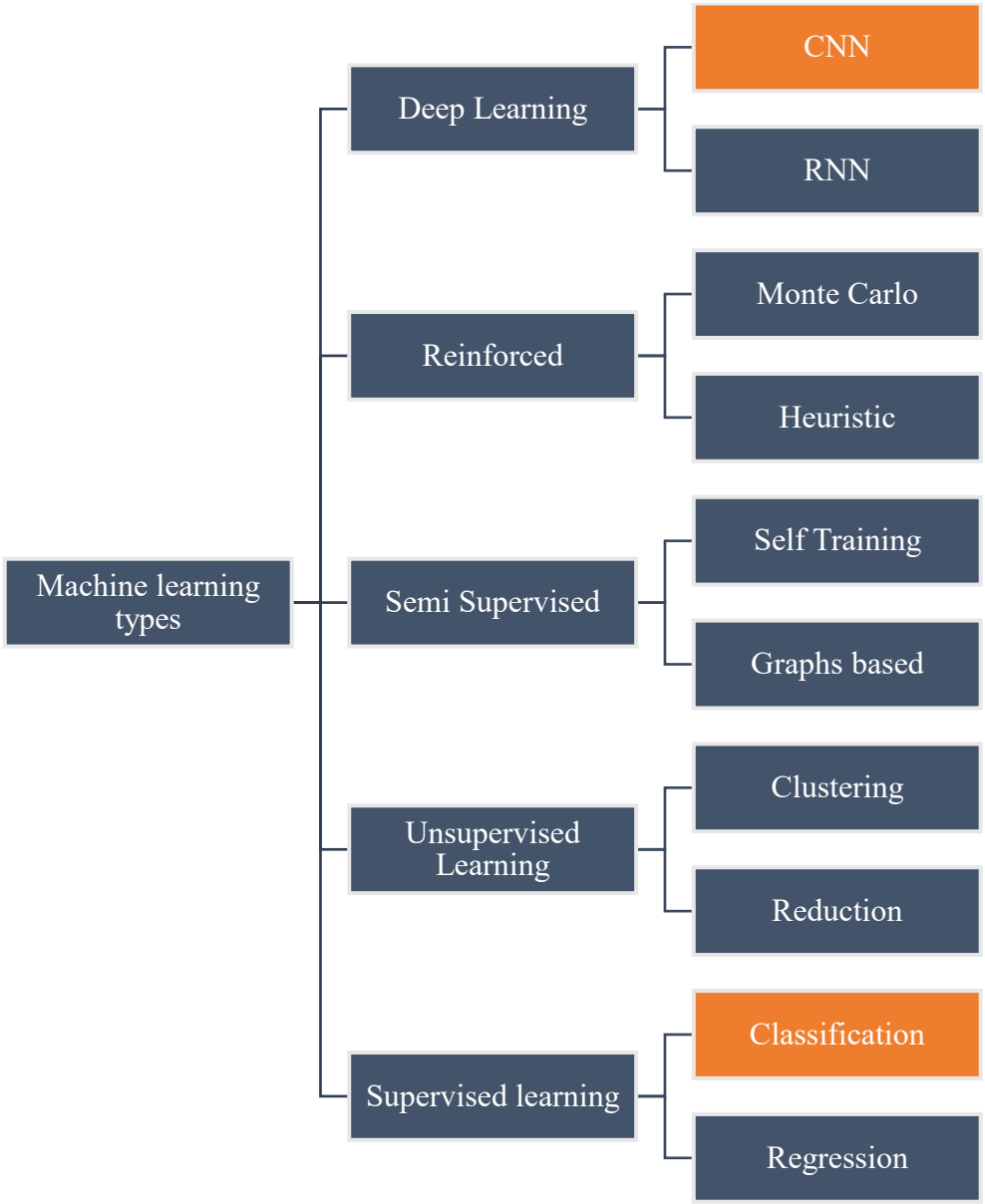
Figure 4.1 Machine Learning Types

Machine learning algorithms are mathematical equations used to perform a specific task. Many machine learning algorithms exist, including linear regression, logistic regression, decision trees, k-nearest neighbor support vector machines, and neural networks. These algorithms can be used for regression, classification, clustering, and dimensionality reduction tasks. The choice of algorithm depends on the task's nature and the data's characteristics.

Many machine learning algorithms are commonly used in remote sensing applications. Some examples include:

- Decision trees: Decision trees are often used in remote sensing for land cover classification and feature extraction tasks. They are easy to interpret and can handle high-dimensional data well.
- Random forests: Random forests are an ensemble learning method that combines multiple decision trees to make a prediction. They are often used in remote sensing for land cover classification and land use mapping. Further will explain in the next chapters.
- Support vector machines (SVMs): SVMs are powerful and versatile algorithms often used in remote sensing for image classification and feature extraction tasks. They work well with high-dimensional data and can handle non-linear relationships between features.
- Neural networks are complex algorithms inspired by the brain's structure and function. They are often used in remote sensing for image classification and interpretation tasks. Convolution neural network is an example used in deep learning here in this study.

- K-means clustering: K-means clustering is an unsupervised learning algorithm often used in remote sensing for image segmentation and land cover classification tasks. It works by dividing the data into k clusters based on similarity.



For this study, Random forests are used. Random forests are a popular choice for many remote sensing applications. Following are the reasons why a random forest might be a better choice than a support vector machine (SVM) for remote sensing tasks:

- Random forests are more robust to noise and outliers in the data. They can handle large amounts of noisy data, which is common in remote sensing applications. In contrast, SVMs can be sensitive to noise and outliers, impacting their performance.
- Random forests are easier to train and tune than SVMs. They do not require careful selection of kernel functions or the setting of complex hyperparameters. This makes them a more practical choice for many remote sensing applications, where the data is often complicated, and there may not be sufficient resources to fine-tune a model.
- Random forests can handle high-dimensional data more efficiently than SVMs. Remote sensing data often has many features, and SVMs can struggle to scale to high dimensions. On the other hand, random forests can handle high-dimensional data more efficiently, making them a better choice for remote sensing tasks.
- Random forests can provide more interpretable results than SVMs. Because random forests comprise many decision trees, it is relatively easy to understand how the model makes predictions. This is not always the case with SVMs, which can be more difficult to interpret.

4.1.1 Structured Data:

The key to remote sensing is to divide your training dataset into training and validation sets, as in any machine learning task. The training set is used to train the model, while the validation set is used to evaluate the model's performance.

- The training set is a subset of the dataset used to fit the model. It consists of input data and corresponding labels (also known as ground truth), and the model uses this data to learn the relationships between the input and the labels.
- The validation set is a separate subset of the overall dataset used to evaluate the model's performance. It consists of input data and corresponding labels, but the model has never seen this data. The model's predictions on the validation set are compared to the ground truth labels to assess its accuracy and generalizability.

By using separate training and validation sets, it is possible to get a more accurate estimate of the model's performance on unseen data. This is important because it helps ensure the model stays within the training data and performs well on new, unseen data. There are a few key rules to follow when splitting your dataset into training and validation sets:

- Use a diverse set of images: A diverse collection for training and validation helps the model generalize better to new, unseen data. This is especially important in remote sensing, where the images can vary significantly due to different types of sensors, resolutions, and environments.
- Balance the classes: If there are multiple classes to classify, it is important to balance the number of examples in each category. For instance, if there are two classes, "forests" and "non-forests," aiming for roughly the same number of examples for each class helps the model classify each class equally well.
- Use stratified sampling: If there is a large dataset to split into training and validation sets, stratified sampling ensures that the distribution of classes is consistent between the two sets. This helps prevent any biases that might be introduced if the data is randomly split.
- Avoid overfitting: When creating the training and validation sets, avoid including similar images. This helps prevent overfitting when the model performs well on the training data but poorly on new, unseen data.
- Consider the context: When working with remote sensing, it's important to consider the circumstances in which the images were taken. For example, ensuring that the data used to train and validate the model includes images taken at different times of the day in different seasons can help the model perform well with new data collected in similar situations.

4.2 System Specifications:

The system specification for training and running the models is as follows:

- 16 GB of RAM
- A 1050 or higher NVIDIA graphics card
- A Core i7 8th generation CPU
- Google Colab

While the above system is considered a standard middle-tier system, it is important to note that the recommended specification for machine learning tasks is generally much higher to achieve optimal performance. Using a higher-end system can improve computational time and accuracy. The performance of different techniques and tools for this project will be compared.

4.3 Software Platforms:

For visualization, Quantum GIS (QGIS), an open-source software, was primarily used, while Orfeo Toolbox was utilized for machine learning within the QGIS interface. However, limited support for deep learning in QGIS led to using TensorFlow and PyCharm to train deep learning models. Trained models were implemented using ArcGIS Pro, which supports pre-trained deep-learning models.

GitHub libraries were also utilized for online access to deep learning models and related tools, providing a platform for hosting and sharing code repositories that facilitated collaboration and reproducibility. OpenCV was used for some of the GitHub libraries, giving access to pre-trained models and tools essential for implementing and evaluating deep learning models for remote sensing applications.

Google Colaboratory (Colab) was also used to access deep learning models and tools online. Colab is a free, cloud based Jupyter notebook environment that enables researchers to write and run Python code online, providing access to powerful hardware resources such as GPUs and TPUs essential for efficiently training deep learning models. Colab supports popular deep learning frameworks like TensorFlow and PyTorch and data visualization and analysis libraries like Matplotlib and Pandas.

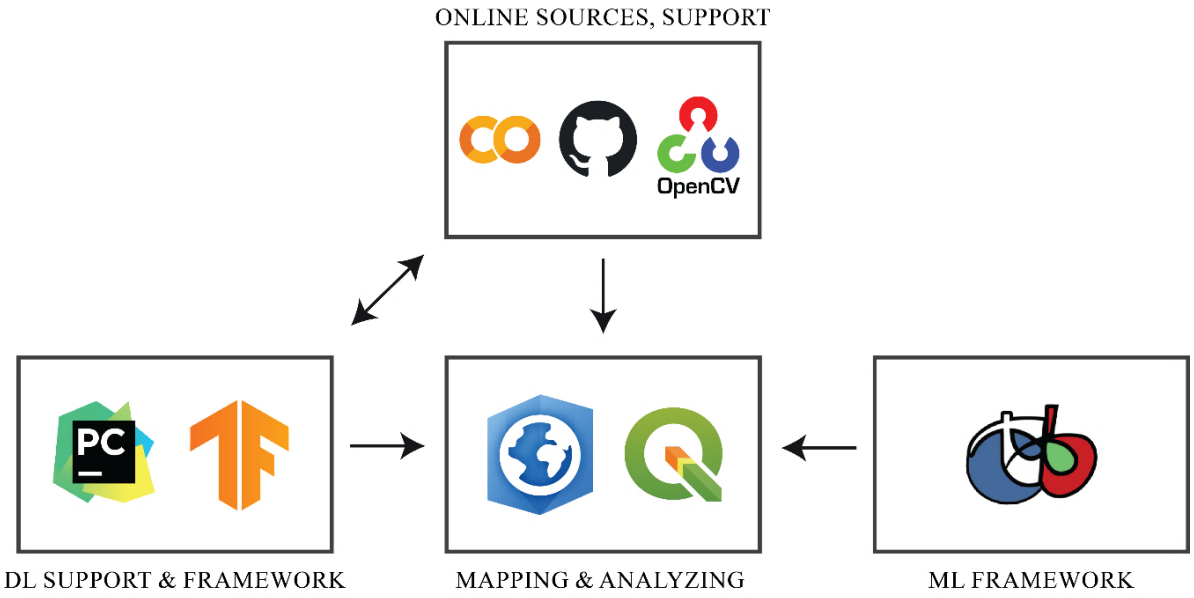
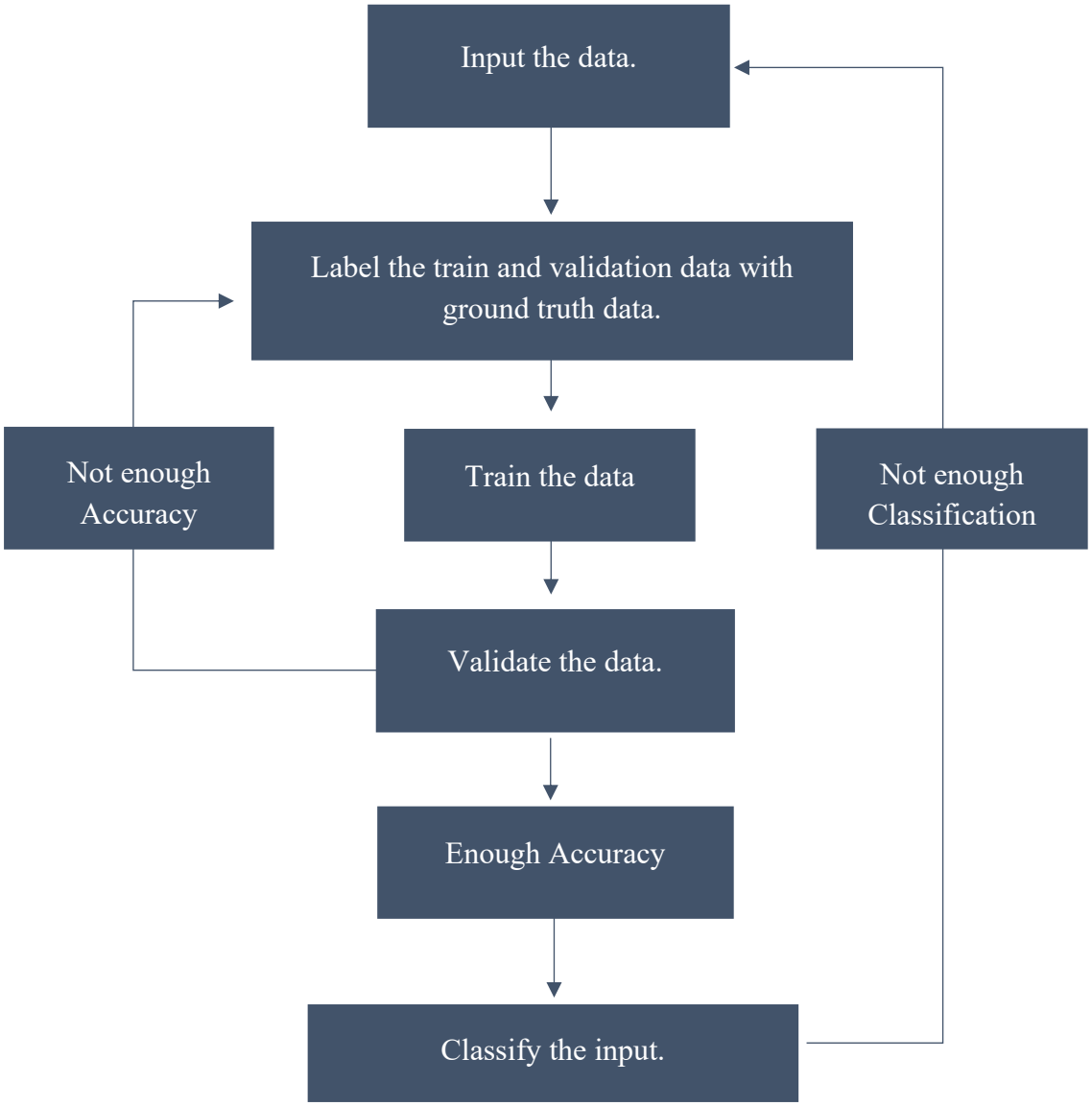


Figure 4.2 Sources & Platforms used for this research study.

4.4 Methodology Flowchart:

This study aimed to classify and extract features from remote sensing data to better understand the study area's characteristics. To achieve this goal, unsupervised classification was initially performed on the primary and integrated datasets to determine the minimum number of classes required for subsequent supervised classification.

Next, supervised classification was conducted on both datasets using pixel-based and object-based methods to compare their performance. Furthermore, a deep learning model was utilized to extract a single feature from large, unstructured data. The deep learning model was applied to both datasets individually and in extended areas to thoroughly investigate the results and identify any patterns or trends in the data.



5 Chapter: Implementation

This chapter will explore the implementation of machine learning techniques, specifically supervised and unsupervised classification, using the methodology discussed in the previous chapter.

Note: Due to limited computational capacity, the machine learning algorithm is only applied to 10% of the urban corridor studied in case 1, covering an area of 2 square kilometers. However, the results are expected to reflect the overall size since the environment remains unchanged.

5.1 Unsupervised classification:

Unsupervised Classification is a machine learning method that can classify data or classes without using labeled training data. In urban environment detection, unsupervised Classification could identify different land cover types or land use within an urban area.

5.1.1 K-means clustering:

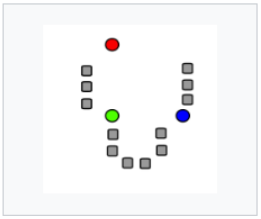
K-means clustering is a popular unsupervised machine learning method that can group data into a predetermined number of clusters. It is often used for data classification tasks, including in the context of urban environment detection.

The k-means algorithm begins by randomly selecting k points to serve as the initial centers of clusters, also known as centroids. Each data point is then assigned to the group with the nearest centroid. Once every moment has been given to a collection, the centroids are recalculated as the average of all the points. This process continues until the centroids do not move or the assignments of points groups remain constant.

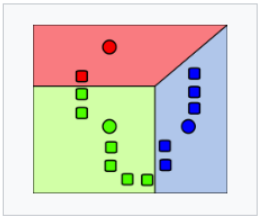
One of the main advantages of the k-means algorithm is that it is relatively fast and efficient, especially for large datasets. However, it can be sensitive to the initial placement of the cluster centers and may only sometimes produce the optimal clusters. It is also limited to dividing the data into a predefined number of groups, which may only sometimes be the best representation of the data. Despite these limitations, k-means Classification is a widely used and effective method for unsupervised classification tasks.

Unsupervised Classification can be useful for identifying data patterns and exploring the relationships between different features. However, it is important to note that unsupervised Classification does not require prior knowledge about the classes in the data, and the resulting classes may not correspond to meaningful categories. As a result, unsupervised clustering is often used as a preliminary step in a larger analysis rather than a standalone method. The following is a demonstration of how k-means work. (*K-Means Clustering - Wikipedia*, n.d.)

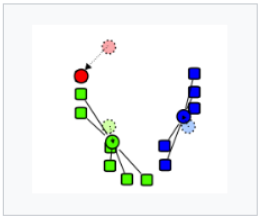
Demonstration of the standard algorithm



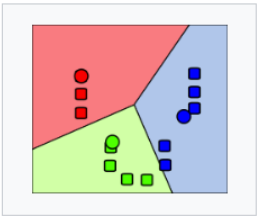
1. k initial "means" (in this case k=3) are randomly generated within the data domain (shown in color).



2. k clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.



3. The centroid of each of the k clusters becomes the new mean.



4. Steps 2 and 3 are repeated until convergence has been reached.

5.1.2 *Implementation on Dataset 1 Primary:*

In this first implementation of unsupervised classification on a primary dataset consisting of only three bands in the RGB format, it was found that at least four classes were needed in the given urban environment for supervised classification. The process was performed without human intervention or structured data, and the processing time was relatively quick. The assessment was based solely on visual inspection.

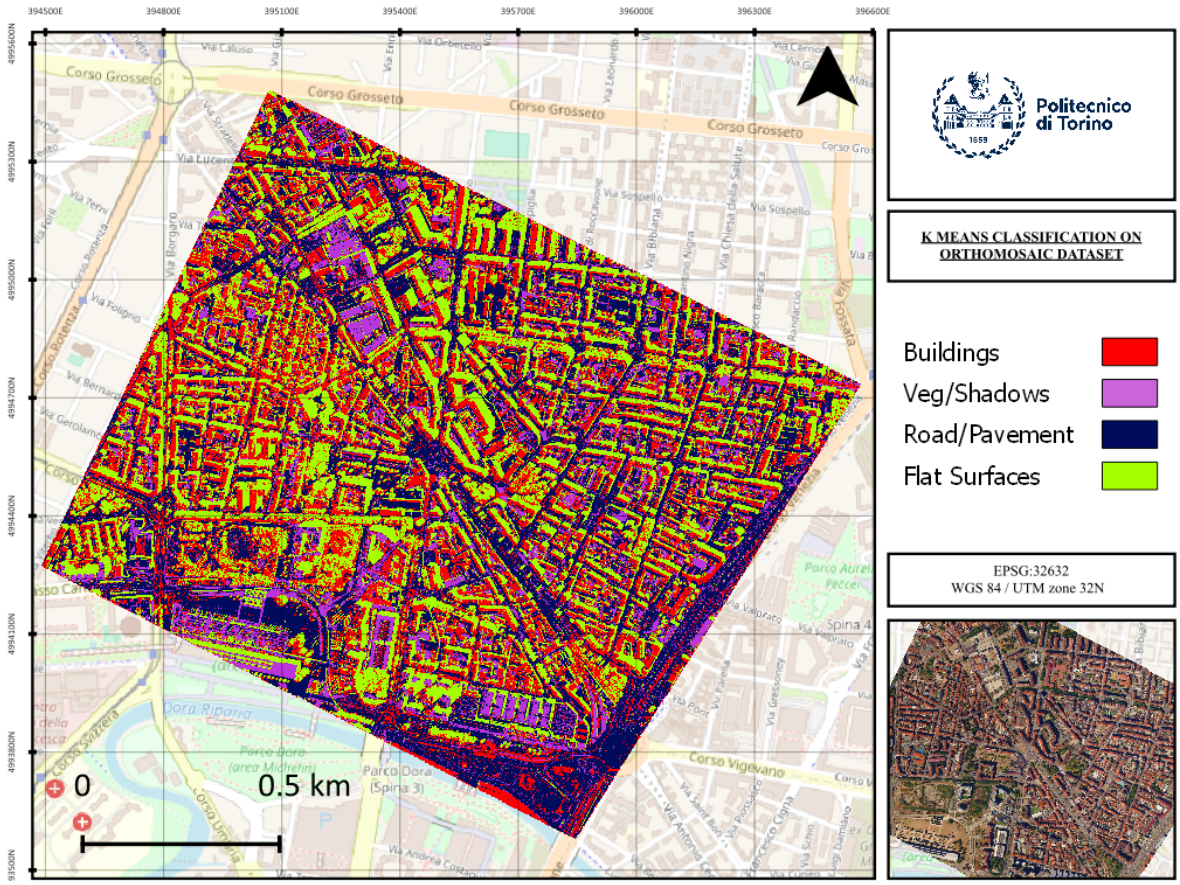


Figure 5.1 Unsupervised Classification Dataset 1

However, due to the limited number of bands used as input (only 3), some pixels needed to be misclassified as either roads or buildings. There were also some false negatives in the vegetation class. In the next step, the analysis was repeated using an additional dataset to make a comparison.

5.1.3 *Implementation of Dataset 2 Integrated:*

This step analyzed the result of unsupervised classification on the integrated dataset. It was observed that the computational time for this dataset was longer than the primary dataset. In the previous analysis using the primary dataset, there was a problem with correctly identifying features with the same radiometric content but different representations.

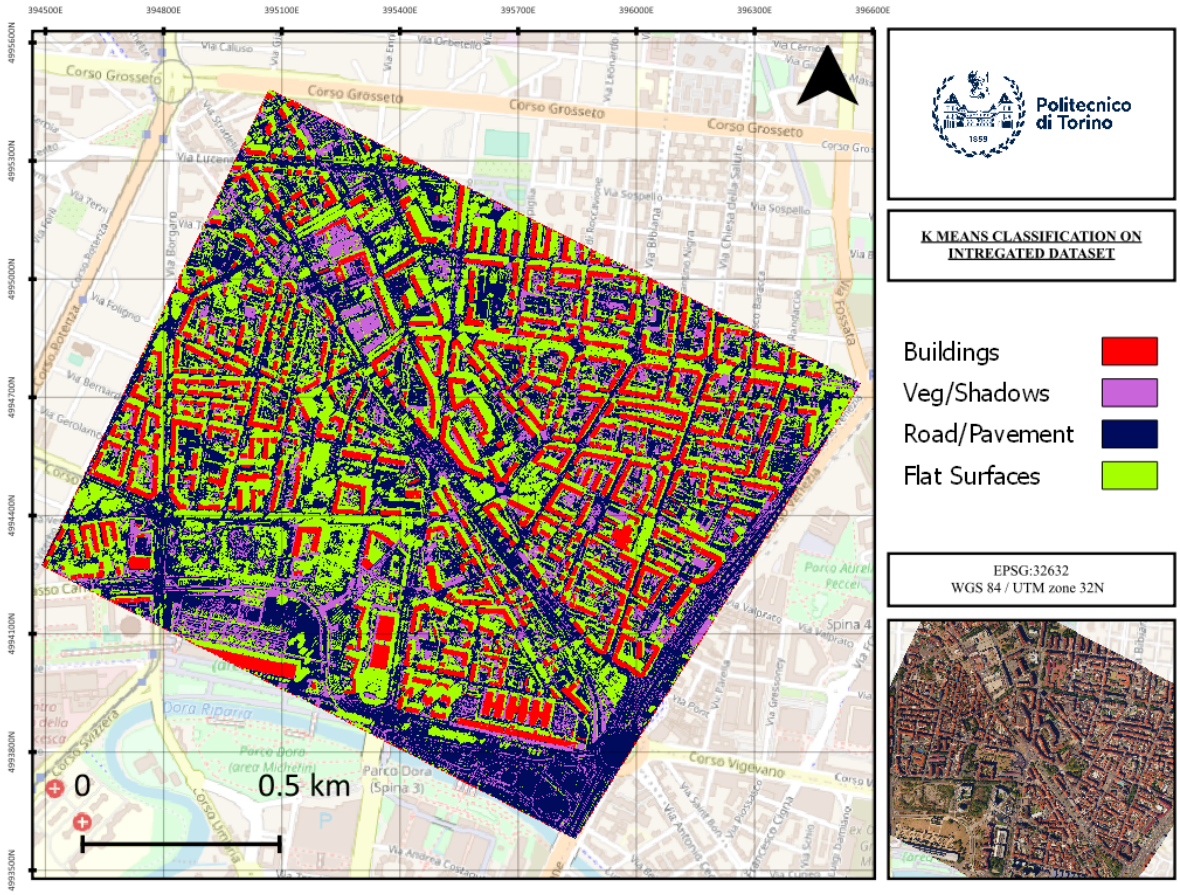


Figure 5.2 Unsupervised Classification Dataset 2

However, by visually inspecting the results of the unsupervised classification on the integrated dataset, it was clear that adding elevation information had significantly reduced the number of errors. This is because the elevation information distinguished between concrete pavement/roads and flat concrete roofs. As a result, the integrated dataset performed much better in the urban environment than the primary dataset. This comparison was also verified in supervised classification analysis.

5.2 Supervised Classification:

Supervised Classification is a type of image classification where the user provides labeled training data to the model, which is used to learn to classify images into predefined classes. The training data consists of pairs of images and corresponding labels, which tell the model what class each image belongs to. Once the model has been trained, it can classify new, unlabeled images.

There are two main types of image classification: pixel-based and object-based.

- Pixel-based Classification involves classifying each pixel in an image based on its characteristics, such as intensity, color, or texture. This type of classification is often used when the objects of interest are well-defined and distant and when the image's resolution is high enough to allow for accurate pixel-level Classification.
- Object-based classification, conversely, involves classifying groups of pixels that belong to the same object or feature. This type of Classification is often used when the things of interest are more complex or the image's resolution is not high enough for pixel-level Classification. In object-based Classification, the model is trained to recognize nine the shape, size, and other characteristics of the objects of interest rather than individual pixels.

Pixel-based and object-based Classifications have their strengths and weaknesses, and which is most appropriate will depend on the specific task and the characteristics of the data. The following sections are the implementation of both types.

5.2.1 Training and Validation Data:

The first step of supervised classification involves creating the training and validation data. There are several ways to do this:

- Collecting field data: Field data can be collected using various methods such as ground surveys, aerial surveys, or satellite data. This data can be used to create training data for remote sensing applications.
- Using existing datasets: Many existing datasets can be used to create training data for remote sensing applications. For example, OpenStreetMap (OSM) is a popular dataset that contains geospatial data.
- Using simulation tools: Simulation tools can create synthetic data for remote sensing applications. This can be useful for testing and development, allowing the creation of data representing various scenarios and conditions.
- Crowdsourcing: Crowdsourcing can be used to create training data for remote sensing applications. For example, platforms like Amazon Mechanical Turk can label satellite images or other data.
- Machine learning techniques: Unsupervised learning techniques can cluster data into different classes, which can be used as training data for a supervised learning algorithm in remote sensing applications.

Ground data and visual control techniques were utilized to create the training and validation data. Figure 5.3 illustrates that this process involved creating validation data using these methods.

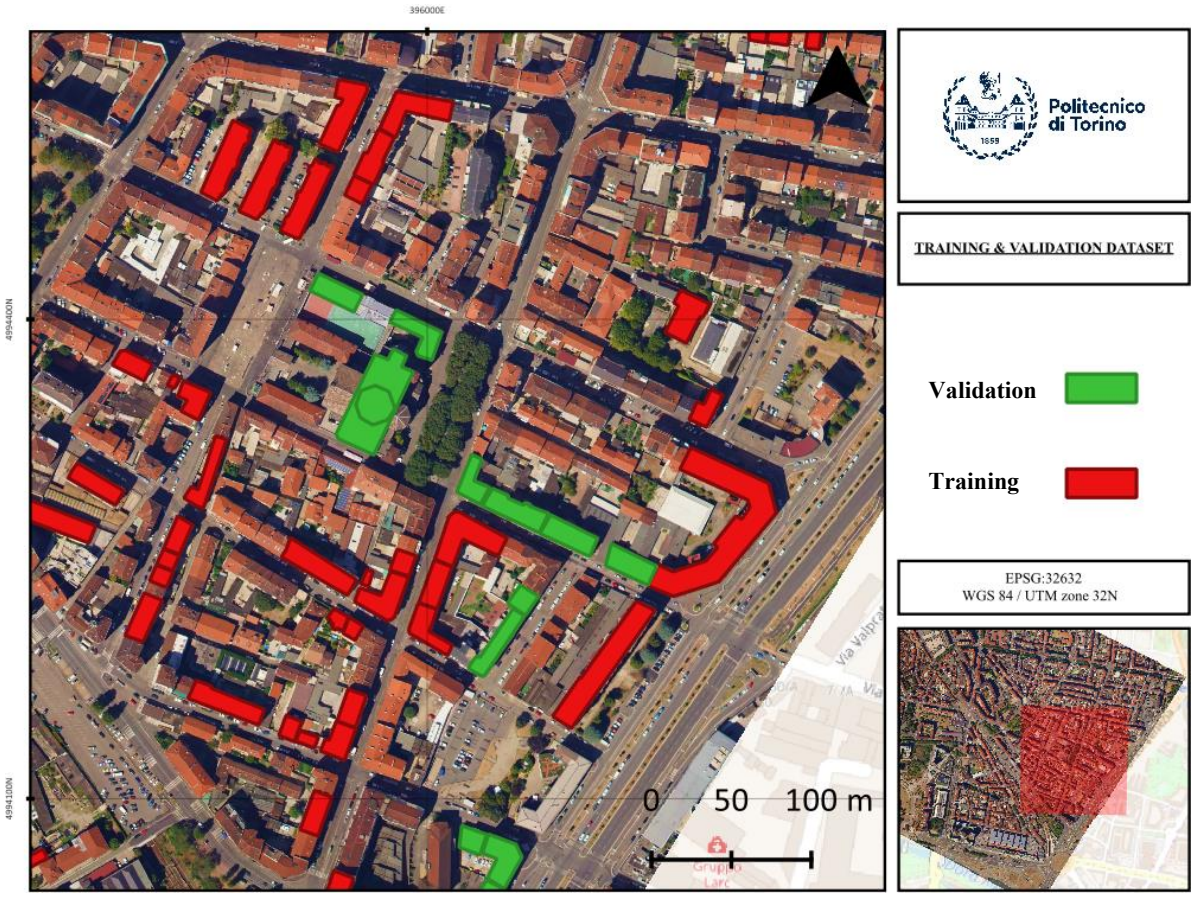


Figure 5.3 Labeling & Training of Data

5.2.2 *Random Forest Algorithm:*

Random forest is a machine learning algorithm that combines the output of multiple decision trees to reach a single result. It is based on the ensemble learning technique of bagging, which creates different training subsets from sample training data with replacement. The final output is based on majority voting for classification and average for regression(*Random Forest Algorithms - Comprehensive Guide With Examples*, n.d.).

A random forest can handle continuous and categorical variables and perform well for classification and regression tasks. It has several advantages over decision trees, such as reducing overfitting, increasing accuracy, and providing feature importance(*What Is Random Forest? | IBM*, n.d.).

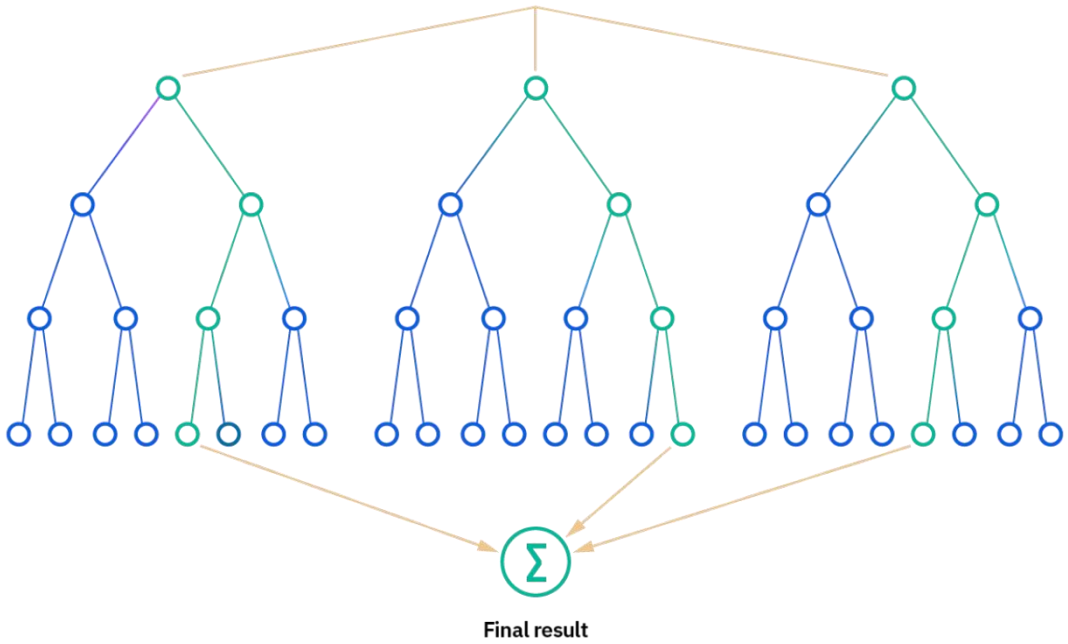


Figure 5.4 RF tree diagram

5.2.2.1 *Original paper*

The original paper on random forest algorithm was published by Leo Breiman in 2001. (*Randomforest2001*, n.d.)

- Random Forest 2001

5.2.2.2 *How does random forest work?*

The basic steps of the random forest algorithm are as follows:

1. Given a training data set, create different bootstrap samples by randomly selecting observations with replacements.
2. For each bootstrap sample, grow a decision tree by recursively splitting the nodes based on a subset of features randomly chosen at each node.
3. Repeat steps 1 and 2 until a desired number of trees are grown in the forest.

4. For classification, predict the class label for a new observation by taking the majority vote of the class labels indicated by each tree in the forest.
5. For regression, predict the numerical value for a new observation by taking the average of the numerical values expected by each tree in the forest.

5.2.2.3 What are the important features of the random forest?

Some of the important features of random forests are:

- It can handle missing values and outliers in the data using the median or mean imputation and robust splitting criteria, respectively.
- It can deal with unbalanced data sets using class weights or balanced bootstrap samples.
- It can reduce the variance and improve the model's generalization by averaging the predictions of multiple trees trained on different subsets of data and features.
- It can estimate the prediction error by using out-of-bag (OOB) samples, which are the observations not included in any bootstrap sample.
- It can measure the importance of each feature by calculating the decrease in accuracy or impurity.

5.2.2.4 How is a random forest different from a decision tree?

A random forest is an extension of a decision tree that uses multiple trees instead of one to make predictions. Some of the differences between random forest and decision tree are:

- Random forest introduces randomness in two ways: bootstrap samples instead of the whole data set and a subset of features instead of all features at each node. This reduces the correlation among the trees and increases the diversity of the forest.
- Random forest reduces overfitting and improves accuracy by averaging the predictions of multiple trees trained on different subsets of data and features. Decision trees tend to overfit the training data and have a high variance when applied to new data.
- The random forest provides feature importance by calculating the decrease in accuracy or impurity when a feature is randomly permuted. A decision tree does not have a direct way to measure feature importance.

5.2.2.5 What are the important hyperparameters in the random forest?

Some of the important hyperparameters in the random forest are:

- Several trees: The number of trees growing in the forest. A larger number of trees can reduce variance, improve accuracy, and increase computation time and memory usage.
- Maximum depth: The maximum depth of each tree. A deeper tree can capture more complex patterns in the data and increase overfitting and variance.
- Minimum samples split: The minimum number of samples required to split an internal node. A larger value can prevent overfitting, reduce variance, and increase bias and underfitting.

- Minimum samples leaf: The minimum number of samples required at a leaf node. A larger value can smooth the prediction, reduce variance, and increase bias and underfitting.

5.2.2.6 Why Random Forest is Better for Aerial and Satellite Imagery

Random forest is a suitable aerial and satellite imagery algorithm for several reasons.

- First, it can handle high-resolution and high-dimensional data common in remote sensing applications.
- Second, it can deal with complex and heterogeneous landscapes with different types of trees, vegetation, and land cover. (T. Zhang et al., 2021)
- Third, it can provide feature importance and explainable identification of trees by using techniques.
- Fourth, it can be easily integrated with other machine learning methods, such as convolutional neural networks, to improve the accuracy and robustness of the classification.

5.3 Pixel-Based Supervised classification:

Pixel-based Classification is a method of image classification in remote sensing that involves assigning a class label to each pixel in an image based on its characteristics. Here are the steps involved in pixel-based Classification:

- Preprocessing: Before the actual Classification can begin, the image must be preprocessed to correct for atmospheric and geometric distortions and enhance the features of interest. This may include correcting for radiometric distortion, removing clouds, and correcting for topographic relief.
- Feature extraction: In this step, the relevant features of each pixel are extracted and transformed into a set of numerical values that can be used to classify the pixels. This may involve calculating texture measures, extracting spectral bands, or estimating indices such as the Normalized Difference Vegetation Index (NDVI).
- Classification: Once the extracted features are, a classification algorithm is applied to the image to assign a class label to each pixel. Different classification algorithms can be used, including maximum likelihood, decision trees, and support vector machines.
- Validation: After the Classification is complete, it is important to validate the results to ensure accuracy. This can be done using various methods, such as visual inspection of the classified image, comparison to reference data, or accuracy assessment using a confusion matrix.
- Post-processing: After the Classification is complete and validated, the classified image may need to be post-processed to smooth out any classification errors and fill in any missing data. This may involve applying filters or interpolation techniques to the classified image.

5.3.1 Pixel-based Classification on Primary Dataset 1:

The hypothesis that the integrated dataset would perform well as it did in unsupervised classification was first tested to compare the effectiveness of pixel-based and object-based classification methods. The pixel-based classification was applied to both the primary and integrated datasets to achieve this.

The result of this classification on the primary dataset is shown in Figure 5.5, illustrating the output of the pixel-based classification method applied to the primary dataset. The pixel-based classification was chosen as the first method of analysis because it is a widely used and well-established technique in remote sensing. By comparing the results of pixel-based classification on both datasets, insight into each method's potential advantages and disadvantages was hoped to be gained.

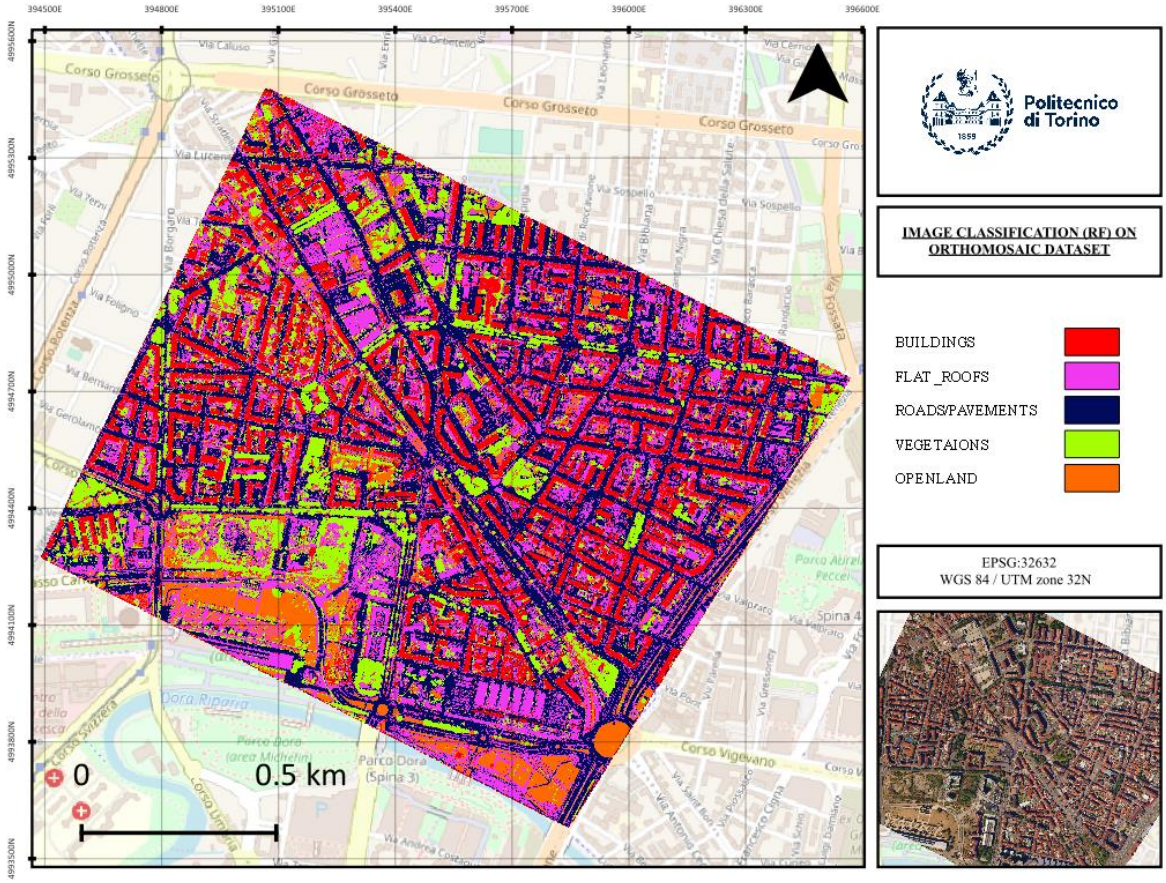


Figure 5.5 Pixel-based classification on Dataset 1

A few points to note about the pixel-based classification results on the primary dataset include the tendency to have more noise, as each pixel is classified individually without considering the relationships between adjacent pixels. However, overall, pixel-based classification accurately organizes features. In this case, lower accuracy was observed in the classification of flat roofs and roads due to their similar radiometric characteristics with only three band values RGB. Additionally, open areas were misclassified as flat roofs at the bottom of the image.

Integration of elevation information into the RGB bands can address these issues and improve classification accuracy. This will provide an additional dimension, namely height, which can be used to define and classify features more accurately. Including this additional information can overcome some of the limitations of pixel-based classification and improve the overall accuracy of the results.

5.3.2 Pixel-based Classification on Integrated Dataset 2:

In this section, the same pixel-based classification method was implemented on the integrated dataset 2, which includes RGB values for each pixel and an additional value representing the elevation of that pixel. With the additional elevation information, features could be distinguished based on their height, which helped to overcome the problem of misclassification observed in the previous classification of the primary dataset. For example, roads and roofs often misclassified due to their similar radiometric characteristics, could now be distinguished based on their different elevations. Figure 5.6 illustrates the result of this classification on the integrated dataset 2, where it can be seen that the inclusion of elevation information significantly improved the accuracy of the classification, particularly in distinguishing roads and roofs.

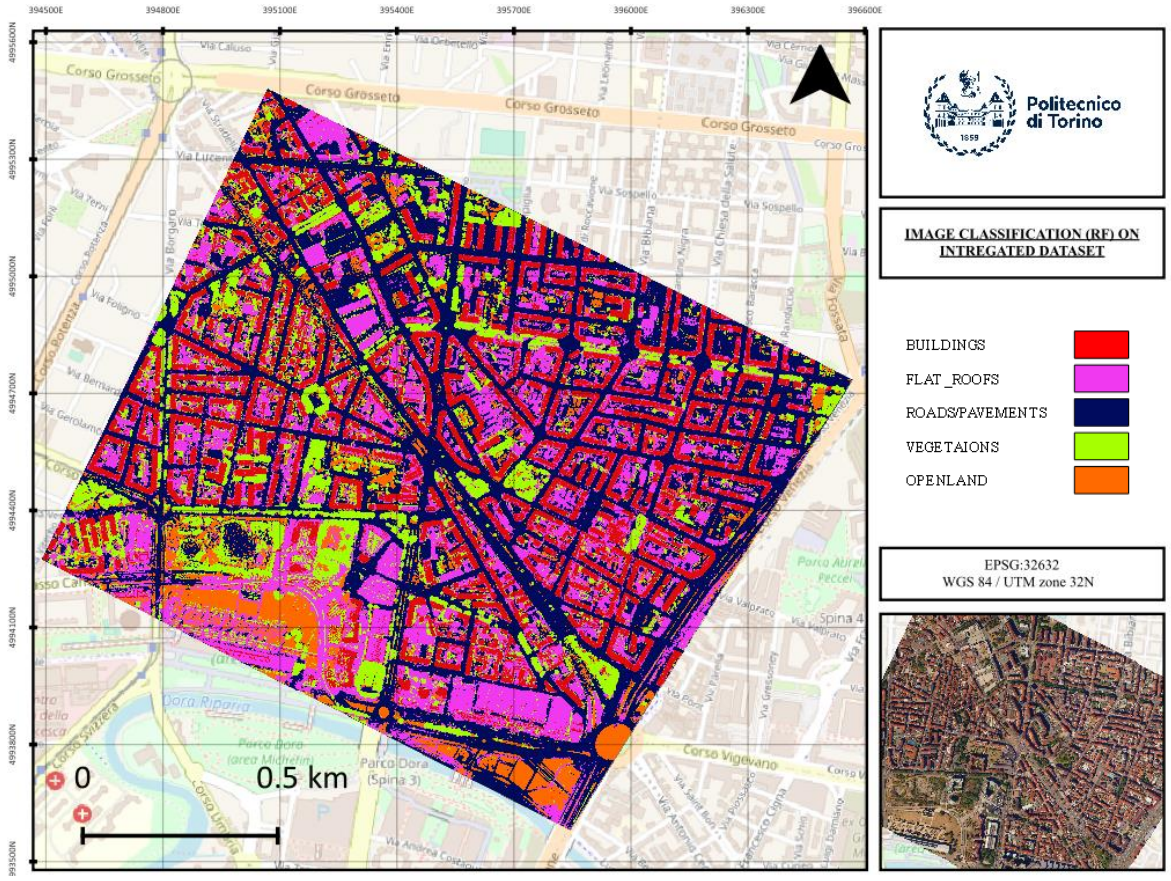


Figure 5.6 Pixel based classification on Dataset 2

5.4 Object-Based Image Classification:

Until this point, it has been determined that the integrated dataset performs well in an urban environment due to including elevation information. Based on this finding, object-based Classification was applied to the integrated dataset 2 to compare the results to those obtained using pixel-based Classification.

Object-based image classification is a method that involves grouping pixels into objects or segments based on their characteristics and then assigning a class label to each object based on the features of the pixels within it. This differs from pixel-based Classification, which assigns class labels to each pixel individually. Object-based Classification can be more accurate because it considers the relationships between adjacent pixels and considers the spatial context of the image.

The process of object-based Classification is like that of pixel-based Classification, with the added step of segmentation, which involves dividing the image into objects or segments based on predefined criteria such as the similarity of pixel values or the proximity of pixels to one another. By grouping pixels into objects, object-based Classification can better account for the spatial relationships between pixels and more accurately classify the image's features.

5.4.1 Segmentation:

In object-based image classification in remote sensing, segmentation divides an image into smaller segments or objects based on predefined criteria. The purpose of segmentation is to group pixels with similar characteristics into things, which can then be classified based on the features of the pixels within them.

Several approaches to segmentation can be used in object-based image classification, including region-based, edge-based, and pixel-based methods.

Region-based methods involve grouping pixels into objects based on the similarity of their pixel values or other characteristics, such as texture or shape. Edge-based methods include identifying and following the boundaries between different features in the image to create segments. Pixel-based methods involve grouping pixels based on their proximity to one another.

Once the image has been segmented into objects, the next step in object-based image classification is to extract the relevant features of each object and apply a classification algorithm to assign a class label to each object.

For this study, the algorithm known as a mean shift is used for segmentation. The mean shift algorithm is a method of clustering and segmentation that can be used in object-based image classification. It works by iteratively shifting the centroids of clusters to the mean location of the points within the cluster until the centroids converge on a stable location. The points are then reassigned to the cluster whose centroid they are closest to. This process is repeated until

the groups have connected and the points within them are homogeneous. Figure 5.7 shows the result of segmentation:

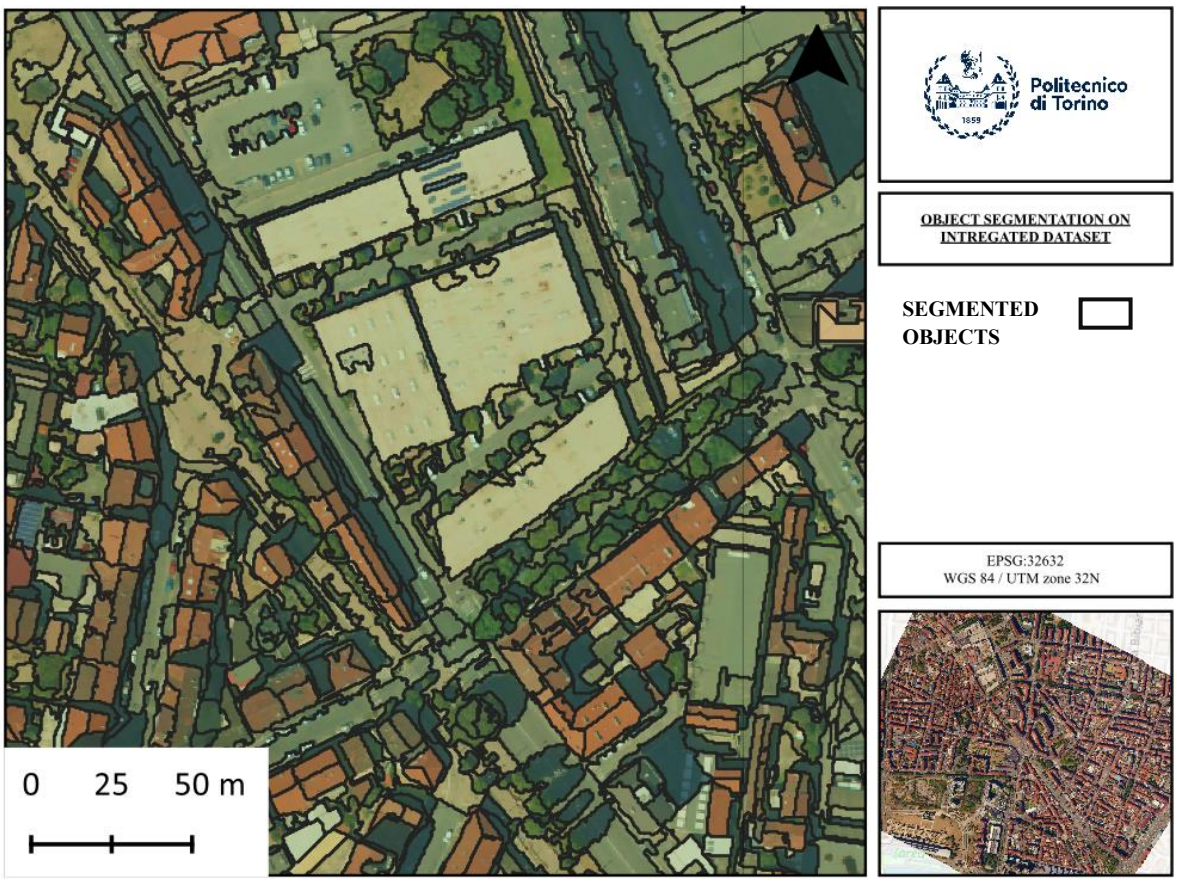


Figure 5.7 Object Segmentation Result on Dataset 2

5.4.2 Object-based Classification on integrated Dataset 2:

Object-based image classification provides several advantages over pixel-based classification. One of the main advantages is its ability to consider the spatial context of the image, which allows it to better differentiate between features that may be difficult to distinguish based on pixel values alone. Additionally, object-based Classification is generally faster than pixel-based Classification once the initial segmentation step has been completed, making it a more efficient method for large datasets.

Object-based classification is particularly beneficial in urban environments, as it can accurately distinguish between complex and varied features. The object-based classification results on the integrated dataset 2 demonstrate this advantage, with less noise in the classified image and more well-defined features as objects. The improved accuracy in distinguishing between roads and roofs, for example, highlights the ability of object-based Classification to differentiate between features based on their shape and size.

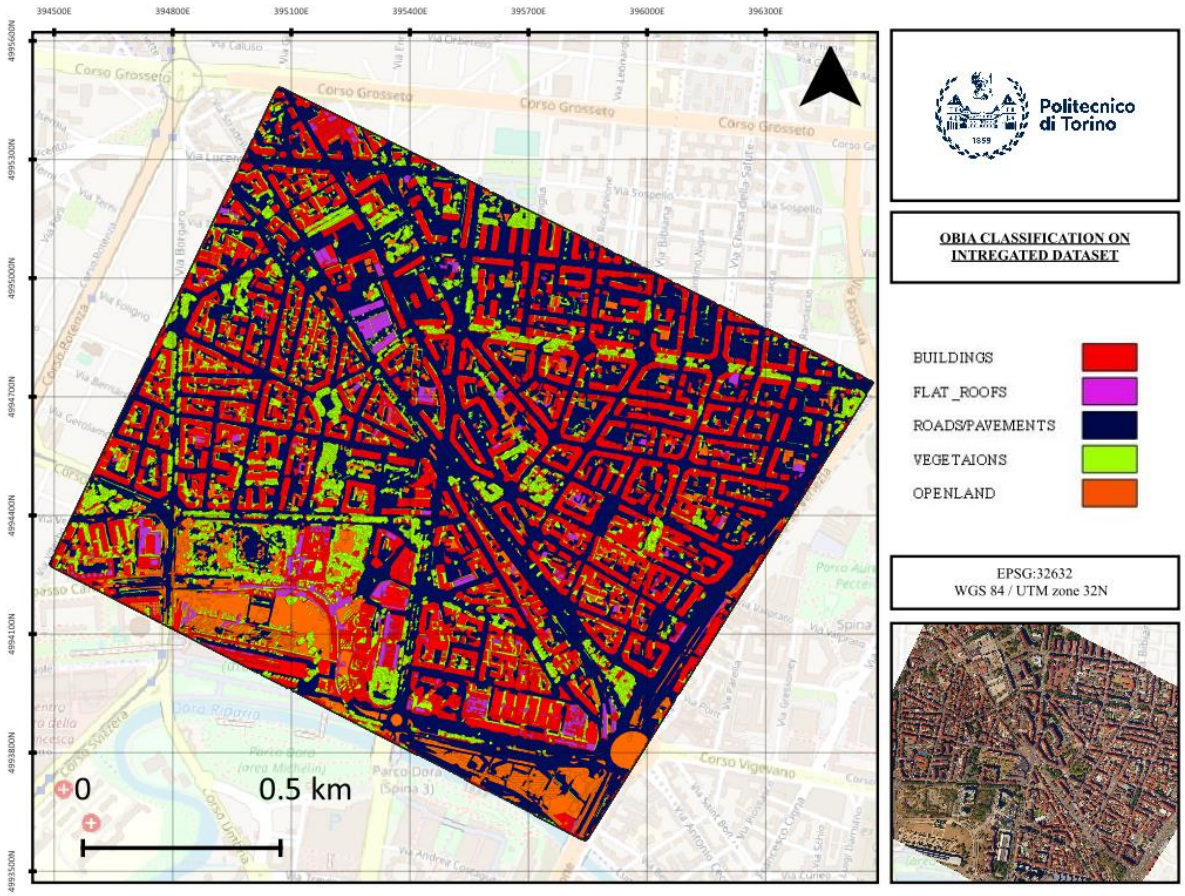


Figure 5.8 OBIA classification On Dataset 2

5.5 Real-life application and artificial neural network:

In a research project focused on using machine learning to identify different features in an urban environment using satellite imagery, objective 4 was to develop and implement a workflow for machine learning in real-life corridor mapping applications. This involves identifying specific elements in a large area, such as white markings on roads.

One challenge with this task is that traditional machine learning algorithms often require structured data with multiple categories to compare and classify objects accurately. However, in this case, the goal is to identify a single element (white road markings). Obtaining sufficiently large and diverse datasets for training a model to recognize this single class can be challenging.

To overcome this challenge, deep learning techniques can be used. Convolutional neural networks (CNNs) are well-suited for image classification tasks and can be effective when working with a single class. These models have many hidden layers and can learn complex data patterns, improving the model's accuracy. Although training a deep learning model may require more time and computational resources, it can produce more accurate results than traditional machine learning approaches when working with a limited dataset.

Once the deep learning model is trained, it can be applied to large areas with similar environments to identify white markings on roads in satellite imagery quickly. This can be a useful workflow for real-life corridor mapping applications where it is necessary to identify this specific feature in a large area accurately. In this case, a deep learning approach can be more effective than traditional machine learning methods.

Deep learning models can also be more accurate than human error for certain tasks, particularly when working with large and complex datasets. However, it is important to note that they may only sometimes be the best choice for a given task and may require more time and computational resources to train. It is important to carefully evaluate the trade-offs between different approaches before deciding which one to use. The study used Convolutional neural networks (CNNs) with pixel classification to achieve objective 4.

5.5.1 U-Net Algorithm:

U-Net is a convolutional neural network that was developed for biomedical image segmentation at the Computer Science Department of the University of Freiburg(Weng & Zhu, 2015). The network is based on a fully convolutional network(*U-Net Explained | Papers With Code*, n.d.. Its architecture was modified and extended to work with fewer training images and yield more precise segmentations. Segmentation of a 512 × 512 image takes less than a second on a modern GPU.

5.5.1.1 Original Paper:

The original paper for the U-Net model is titled U-Net: Convolutional Networks for Biomedical Image Segmentation by Olaf Ronneberger, Philipp Fischer, and Thomas Brox1. It was submitted on 18 May 2015 and accepted at MICCAI 2015. (Weng & Zhu, 2015)

5.5.1.2 How does U-Net work?

The U-Net algorithm works as follows:

- The network consists of a contracting path and an expansive path.
- The contracting path is a standard convolutional network that applies repeated 3x3 convolutions, each followed by a ReLU activation and a 2x2 max pooling operation for downsampling.
- The number of feature channels is doubled at each downsampling step.
- The expansive path involves upsampling the feature map and applying a 2x2 "up-convolution" that halves the number of feature channels.
- The upsampling is followed by concatenation with the corresponding cropped feature map from the contracting path, two 3x3 convolutions, and ReLU activations.
- Cropping is necessary to compensate for the loss of border pixels in every convolution.
- The final layer of the network uses a 1x1 convolution to map each 64-component feature vector to the desired number of classes.
- The network only uses the right part of each convolution without any fully connected layers.
- To predict pixels in the border region of the image, the missing context is extrapolated by mirroring the input image.
- This tiling strategy allows the network to be applied to large images, avoiding limitations in resolution due to GPU memory.

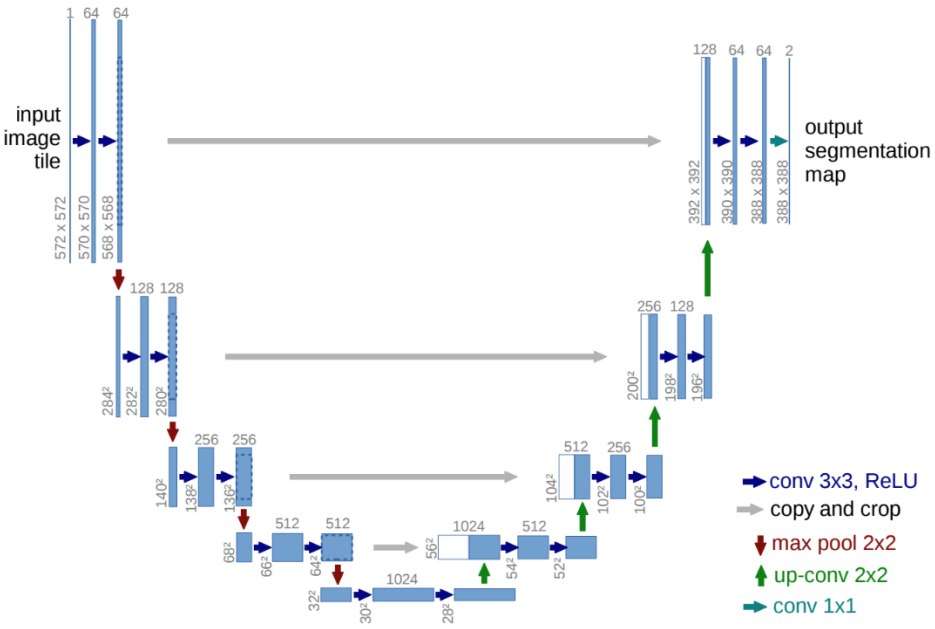


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

Figure 5.9 U-Net Architecture explains in the original paper (Weng & Zhu, 2015)

5.5.1.3 Pseudo-code for using u-net with ResNet34 as backbone:

```
# Import necessary libraries
import torch
import torchvision
import torch.nn.functional as F
from torch import nn

# Define ResNet34 as backbone
backbone = torchvision.models.resnet34(pretrained=True)
# Define model
model = UNet(backbone='backbone', encoder_weights='imagenet')

# Define U-Net architecture
class UNet(nn.Module):
    def __init__(self, in_channels=3, out_channels=1):
        super(UNet, self).__init__()

        # Define encoder (ResNet34)
        self.encoder = nn.Sequential(*list(backbone.children())[:-2])

        # Define decoder
        self.decoder = nn.ModuleList([
            nn.ConvTranspose2d(512, 256, kernel_size=2, stride=2),
            nn.ConvTranspose2d(256, 128, kernel_size=2, stride=2),
            nn.ConvTranspose2d(128, 64, kernel_size=2, stride=2),
            nn.ConvTranspose2d(64, 32, kernel_size=2, stride=2),
        ])

        # Define output layer
        self.output = nn.Conv2d(32, out_channels, kernel_size=1)

    def forward(self, x):
        # Encoder
        enc1 = self.encoder[0](x)
        enc2 = self.encoder[1](F.relu(enc1))
        enc3 = self.encoder[2](F.relu(enc2))
        enc4 = self.encoder[3](F.relu(enc3))

        # Decoder
        dec1 = self.decoder[0](enc4)
        dec2 = self.decoder[1](F.relu(torch.cat([enc3, dec1], dim=1)))
        dec3 = self.decoder[2](F.relu(torch.cat([enc2, dec2], dim=1)))
        dec4 = self.decoder[3](F.relu(torch.cat([enc1, dec3], dim=1)))

        # Output layer
        output = self.output(dec4)

        return output

# Define loss function
criterion = nn.BCEWithLogitsLoss()

# Define optimizer
optimizer = torch.optim.Adam(model.parameters(), lr=1e-4)

# Train model
for epoch in range(num_epochs):
    for images, labels in dataloader:
        images = images.to(device)
```

```
        labels = labels.to(device)

        # Forward pass
        outputs = model(images)
        loss = criterion(outputs, labels)

        # Backward and optimize
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

        # Print loss
        print(f"Epoch [{epoch+1}/{num_epochs}], Loss: {loss.item():.4f}"))

……..
```

5.5.1.4 Why U-Net is Better for Aerial and Satellite Imagery

U-Net is better for aerial and satellite imagery because it can handle images of different sizes and aspect ratios and learn from a few training images(*U-Net for Semantic Segmentation on Unbalanced Aerial Imagery | by Amirhossein Heydarian | Towards Data Science*, n.d.). U-Net can also cope with unbalanced classes and noisy labels using appropriate loss functions such as focal loss or mean intersection over union (mIoU). U-Net can segment aerial and satellite images into various classes: water, land, road, building, vegetation, and unlabeled. U-Net can also be modified and extended to suit different tasks and challenges in aerial and satellite image analysis. (*Satellite Image Segmentation: A Workflow with U-Net | by Vooban | Vooban AI | Medium*, n.d.)

A backbone is a pre-trained network that provides the encoder part of the U-Net, while the decoder is usually custom-built. In this study, the resnet34 model is the backbone of U-Net Architecture in this research.

Using ResNet as a backbone for U-Net can have several advantages:

- ResNet is a powerful, deep network that has achieved state-of-the-art results on various image classification tasks. It can extract rich and high-level features from the input images to help the segmentation performance.
- ResNet has residual connections that allow information to flow across layers without degradation. This can prevent the vanishing gradient problem and enable deeper networks to be trained effectively.
- ResNet's bottleneck structure reduces the parameters and computations in each residual block. This can improve the efficiency and scalability of the network.
- ResNet can be easily adapted to different input sizes and output classes by changing the number of layers and channels. This can make it suitable for various segmentation problems. (*U-Nets with ResNet Encoders and Cross Connections | by Christopher Thomas BSc Hons. MIAP | Towards Data Science*, n.d.)

5.6 Pixel-based Classification using Convolution neural networks:

To perform pixel-level Classification using a CNN(U-Net), the input data is typically a multi-band raster image, such as a satellite image. The output is a label for each pixel indicating its class. CNN is trained on a labeled dataset of ideas, where the class labels for each pixel are known. The model learns to recognize the characteristics of each class in the training data and uses that knowledge to classify new, unseen images.

One advantage of using CNN for pixel-level Classification is that it can learn to recognize patterns at different scales and contexts. This is because CNN(U-Net) uses convolutional layers, which filter input data to extract features at different scales. This can be particularly useful in remote sensing applications, where the size and context of features can vary significantly across an image. This analysis presents two examples of using convolutional neural networks (CNNs) for corridor mapping:

- The first study was conducted in an urban environment in Turin, Italy, and focused on comparing the performance of machine learning and deep learning techniques over a 10x area with similar parameters. This study aimed to evaluate the effectiveness of using CNNs for identifying features in an urban environment using satellite imagery.
- The second investigation was carried out in a road corridor to identify white markings on the entry and exit ramps of a 54.5 km motorway. This study examined a real-life application of using CNNs for corridor mapping and allowed for the analysis of temporal data to understand how the features changed over time.

Overall, these studies demonstrated the potential of using CNN(U-Net) for corridor mapping in urban environments and highlighted the importance of considering the specific requirements of a problem when selecting an appropriate machine-learning approach.

5.6.1 Urban Corridor:

Previously, a study used machine learning techniques to classify a 2 km^2 (10% of the original area) with multiple classes, which took several hours to complete. In contrast, using a convolutional neural network (CNN) and a larger dataset spanning 22.9 km^2, it was possible to classify only one feature in a shorter amount of time (10 minutes) with a higher accuracy (92.08%). The dataset used in this case was the integrated dataset.

However, it should be noted that the CNN model required a longer training time (6 hours) compared to the machine learning approach. Once the CNN model is trained, it can be used to classify similar areas, such as in Italy, with a high degree of accuracy.

Overall, this example demonstrates the potential of using CNNs for efficient and accurate Classification of urban environments in remote sensing applications. While training may require more time and resources, the resulting model can be applied to large areas relatively quickly and achieve higher accuracy than traditional machine learning approaches.

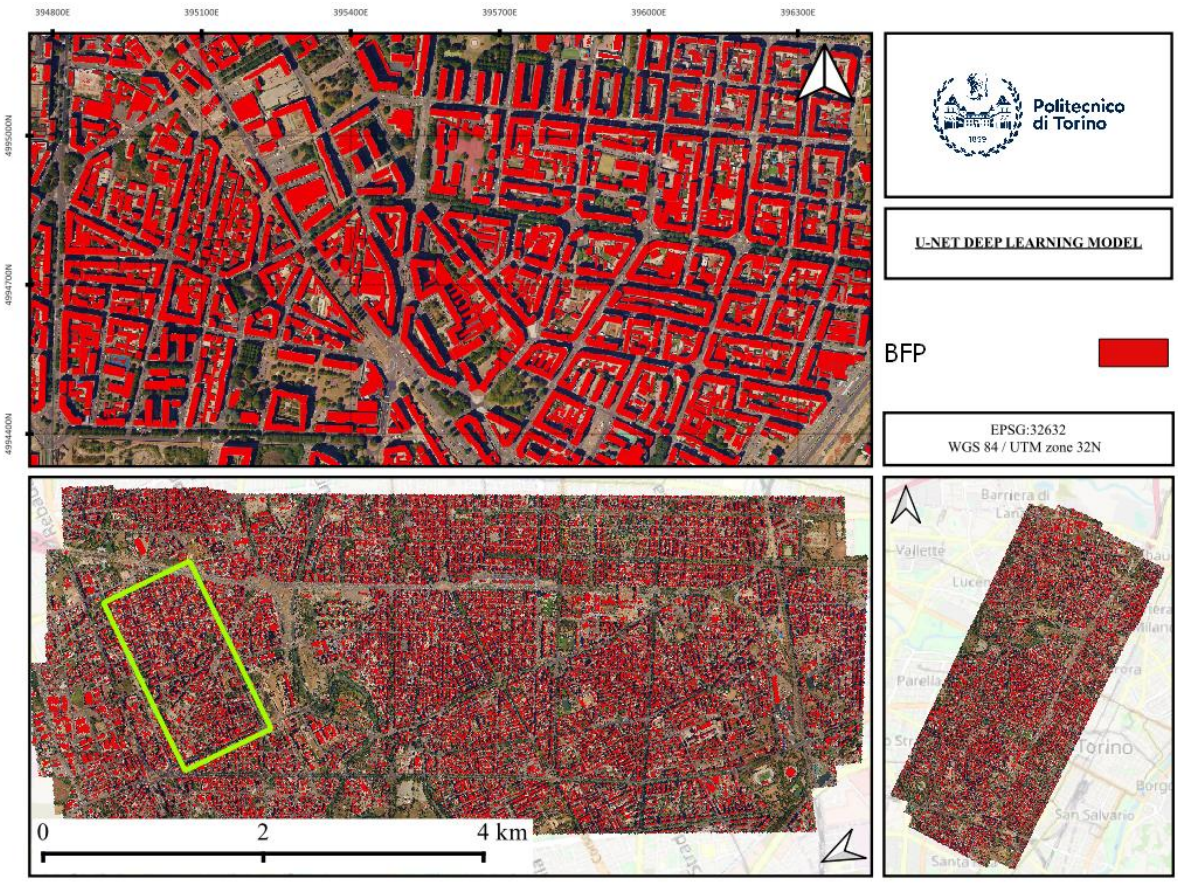


Figure 5.10 U-Net DL Classification on Case Study 1 Urban Corridor

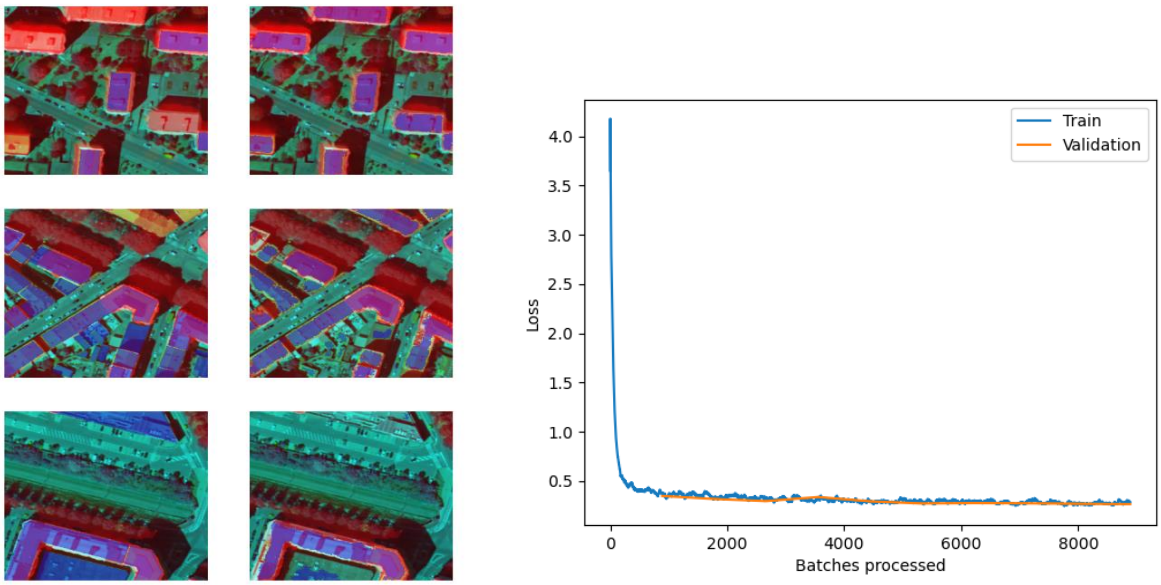


Figure 5.11 DL Model Stabilization

5.6.2 Road Corridor:

The results of using the trained U-Net model on entry and exit ramps are shown here, along with the ground truth and prediction. Below figure 5.12 shows that the white markings can be accurately identified without elevation data, indicating that the primary dataset was sufficient for this task. The accuracy obtained was 91.11%.

The ability to classify only one feature with high accuracy using a U-Net opens a range of potential applications in real-life scenarios, such as identifying pavement cracks or marking roads as per objective 4 of the study. Keeping the basic parameters of the investigations consistent over time makes it possible to make more accurate comparisons and track changes in the features being studied.

Overall, this example illustrates the effectiveness of using CNNs for identifying specific features in remote sensing applications and the potential for using these models to support a variety of real-life tasks.

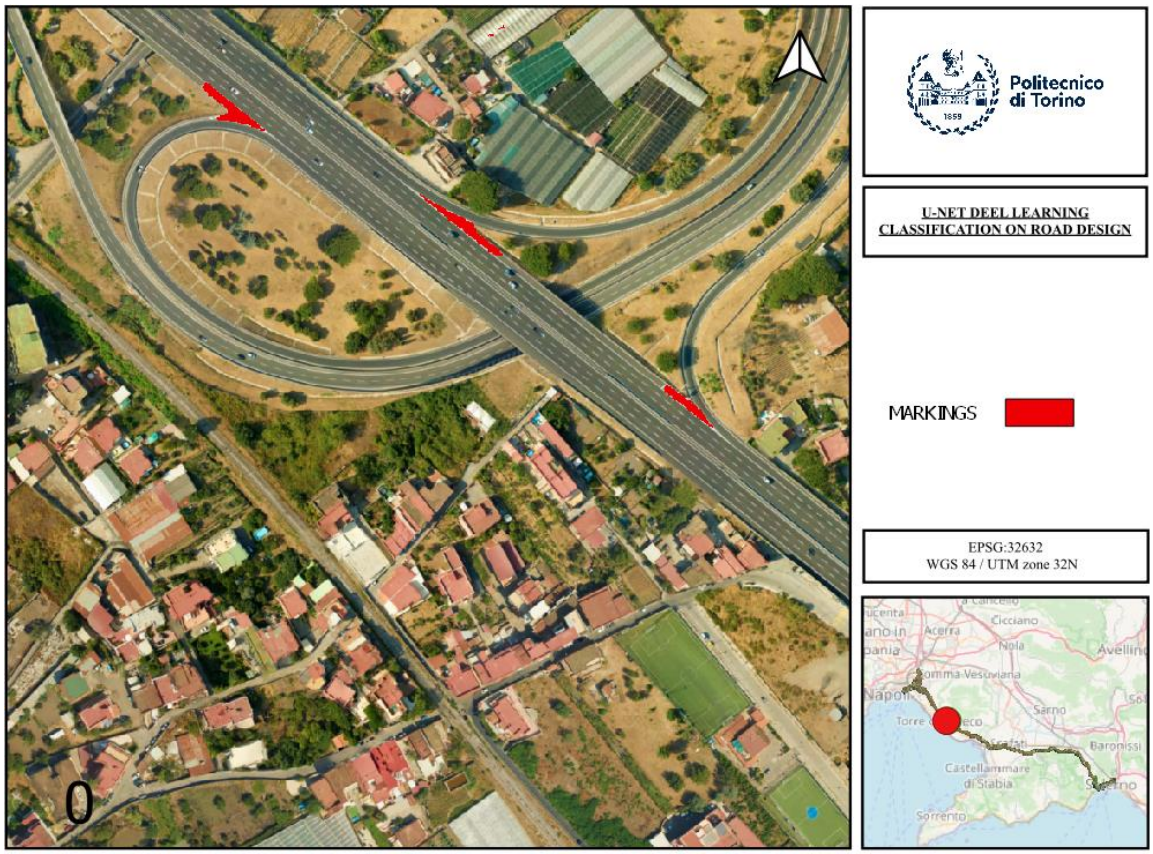


Figure 5.12 U-Net DL Classification on Case Study 2 Road Corridor

6 Chapter: Accuracy Comparisons

So far, supervised and unsupervised classifications have been carried out using various methods and tools. A range of accuracy assessment tools specifically designed for machine learning will now be used to assess the impact of adding additional data bands and compare the performance of pixel-based and object-based classification methods.

6.1 Accuracy Assessments tools:

The specific tools that will be used will depend on the objectives of the study, which are as follows:

- To assess the impact of adding additional data bands on the Classification of urban features in corridor mapping: To address this objective, the classification results on the primary dataset (Dataset 1) will be compared with those obtained using the integrated dataset (Dataset 2).
- To compare the performance of pixel-based and object-based classification methods in corridor mapping: To address this objective, the results of image-based Classification (using either pixel-based or object-based methods) will be compared with those obtained using object-based Classification.
- A visual inspection of the classified images has already been performed. Still, now more objective, quantitative tools will be used to verify the goals and assess the accuracy of the classification methods. Some common methods include:
- Confusion matrix: A confusion matrix is a table that compares a classification model's predicted labels with the data's true labels. It can be used to evaluate the overall accuracy of the model and the specific types of errors it makes.
- Overall accuracy: This is the proportion of the total number of pixels classified correctly by the model. It is a simple but useful metric that can provide a general sense of the model's performance.
- Cohen's kappa coefficient: This statistic measures the agreement between the predicted labels of a classification model and the true labels of the data, considering the possibility of chance agreement. It can be used to assess the reliability of the model's predictions.

Further elaboration on the concept of confusion matrix will be provided as it is the main assessment criterion in this study.

6.1.1 Confusion Matrix:

In remote sensing using machine learning, a confusion matrix is a table that compares a classification model's predicted labels with the data's true labels. It is a useful tool for evaluating the performance of a classification model and can provide insight into the specific types of errors that the model is making.

A confusion matrix is typically organized as follows:

- The matrix rows represent the data's true labels, while the columns represent the predicted labels.
- The matrix cells contain the counts or proportions of data points with a given true and predicted label.

For example, consider a classification model to classify land cover types into forests, grasslands, and urban areas. A confusion matrix for this model might look like this:

Table 6.1 Confusion Matrix Example

	Predicted: Forest	**Predicted: Grassland**	**Predicted: Urban**
True: Forest	50	10	5
True: Grassland	15	50	10
True: Urban	5	10	50

In this example, the model correctly classified 50 data points as forests, 50 as grasslands, and 50 as urban areas. It also made some errors, such as misclassifying 10 data points as grasslands when they were forests and 5 data points as urban when they were forests.

By analyzing the confusion matrix, one can get a sense of the overall accuracy of the model, as well as the types of errors it makes. For example, the model may perform well but needs to distinguish forests from grasslands. This information can be used to improve the model or to choose a different classification method.

6.1.2 Cohen's kappa coefficient:

Cohen's kappa coefficient (also known as the kappa index or simply kappa) is a statistic that measures the agreement between the predicted labels of a classification model and the true labels of the data, considering the possibility of chance agreement. It is a useful tool for assessing the reliability of the model's predictions. The kappa coefficient is calculated as follows:

Kappa = (observed agreement - expected agreement) / (1 - expected agreement)

Where:

- The observed agreement is the proportion of predictions that match the true labels.
- The expected agreement is the proportion of predictions that would be expected to match the true labels by chance based on the distribution of labels in the data.

The kappa coefficient is a metric that ranges from -1 to 1 and measures the agreement between predicted and true labels. A kappa coefficient of 1 means that the predictions and true labels perfectly match, and a value of 0 indicates that the deal is no better than random chance. Negative values indicate that the agreement is worse than random chance.

To interpret the kappa coefficient, it is often useful to compare it to a benchmark value, such as the agreement that would be expected between two independent raters. For example, a kappa value of 0.6 might be considered a good agreement if the agreement between two independent raters is 0.4 but poor if the expected agreement is 0.8.

6.1.3 User and Producer Accuracy:

User and producer accuracy measures a classification model's accuracy in remote sensing. They are used to evaluate the model's performance and identify potential sources of error.

- User accuracy is the proportion of data points correctly classified by the model relative to the total number of data points. It measures the model's overall accuracy and is typically the most important metric when evaluating the performance of a classification model.
- Producer accuracy is the proportion of data points belonging to a particular class correctly classified by the model relative to the total number of data points belonging to that class. It measures the model's accuracy for each class individually and can be useful for identifying any biases or patterns in the model's performance.

For example, consider a classification model to classify land cover types into forests, grasslands, and urban areas. The user accuracy of the model might be 0.8, meaning that 80% of the data points were correctly classified overall. The producer accuracy for each class might be 0.9 for forests, 0.7 for grasslands, and 0.6 for urban areas, indicating that the model performs better for forests than for the other classes.

6.2 Objective 2: Comparison of primary and integrated Dataset

The accuracy assessment results indicate that the integrated dataset (Dataset 2) improves classification performance compared to the primary dataset (Dataset 1). This can be seen in the confusion matrices for the two datasets, which show that the overall accuracy of the Classification increased from 0.83 to 0.89, and the kappa index, a statistical measure of agreement between the predicted and true labels, rose from 0.79 to 0.87.

A closer examination of the confusion matrices reveals that including elevation data in the integrated dataset leads to particularly significant improvements in the classification of flat roofs and roads. For example, the user accuracy for the classification of flat roofs increased from 0.59 in Dataset 1 to 0.77 in Dataset 2, while the user accuracy for the classification of roads improved from 0.79 to 0.94. This suggests that the additional dimension of elevation data helps to better distinguish between these features in the urban environment.

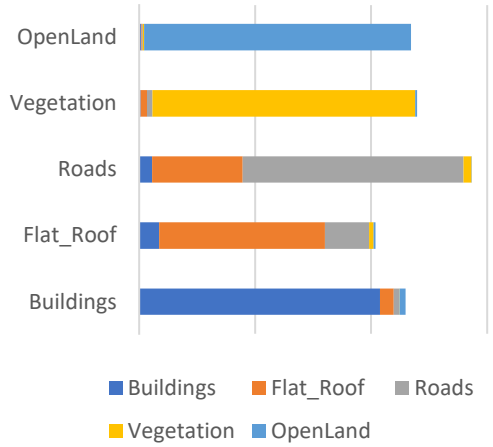
Table 6.2 Dataset 1 Confusion Matrix

Primary DataSet:1	**User Accuracy**	**Producer Accuracy**
Buildings	0.869	0.904
Flat Roof	0.598	0.7
Roads	0.796	0.664
Vegetation	0.948	0.945
Open Land	0.964	0.982
	Overall Accuracy	**0.835**
	Kappa index	0.793

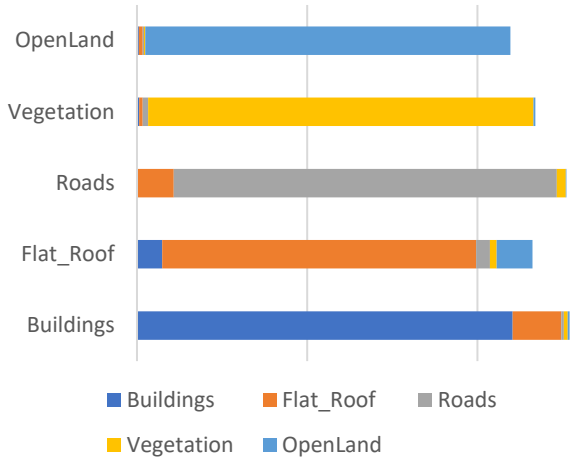
Table 6.3 Dataset 2 Confusion Matrix

Integrated DataSet:2	**User Accuracy**	**Producer Accuracy**
Buildings	0.926	0.868
Flat Roof	0.773	0.793
Roads	0.944	0.892
Vegetation	0.949	0.967
Open Land	0.899	0.978
	Overall Accuracy	**0.898**
	Kappa Index	0.873

Primary Dataset: 1 (83% Accuracy)



Integrated Dataset:2 (89.9% Accuracy)



The bar graph comparison shows that using the integrated dataset (Dataset 2) improves classification performance for flat roofs and roads compared to the primary dataset (Dataset 1). In particular, the integrated dataset significantly reduces the misclassification of flat roofs as roads and vice versa.

These results support the conclusion that the integrated dataset performs well in corridor mapping and that object-based Classification is superior to pixel-based Classification. Based on these findings, Proceeding to the second part of the supervised Classification, object-based image classification.

6.3 Objective 3: Comparison of Pixel-based vs. Object-based image classification

As part of the third objective, to compare the performance of pixel-based and object-based classification methods in the urban environment, the integrated dataset (Dataset 2) was used, established in the second objective as being well-suited for this purpose. The results of the comparison are shown in the tables below.

Table 6.4 Pixel-Based Classification: Confusion Matrix

Pixel Based	**User Accuracy**	**Producer Accuracy**
Buildings	0.926	0.868
Flat Roof	0.773	0.793
Roads	0.944	0.892
Vegetation	0.949	0.967
Open Land	0.899	0.978
	Overall, Acc.	**0.898**
	Kappa Index	0.873

Table 6.5 Object Based Classification: Confusion Matrix

Object based	**User Accuracy**	**Producer Accuracy**
Buildings	0.984	0.863
Flat Roof	0.786	0.982
Roads	0.936	0.927
Vegetation	0.92	0.963
Open Land	0.957	0.938
	Overall, Acc.	**0.924**
	Kappa index:	0.921

Overall, the results indicate that object-based Classification leads to a slight improvement in accuracy compared to pixel-based Classification, increasing from 89.8% to 92.4%. While this increase is not necessarily statistically significant, it does suggest that object-based Classification is a robust and reliable method for classifying features in the urban environment.

It is worth noting that object-based Classification requires an additional step, namely the segmentation of the image into objects or regions, and may also require more human intervention in the form of manual labeling or editing of the classified image. However, the benefits of object-based Classification, including the ability to consider the spatial context of

the image and to account for the relationships between adjacent pixels, may make it a worthwhile investment of time and effort in certain applications.

6.4 Objective 4: Deep Learning Results on Corridor Mapping

The fourth objective of this study was to evaluate the use of machine learning and deep learning algorithms for corridor mapping in real-life applications. One concern for traditional machine learning algorithms is their inability to accurately identify specific features or classes with high accuracy, often requiring significant human intervention and resulting in a large amount of redundant data. To address this issue, the study incorporated deep learning techniques in the later stages of the investigation. The key findings from this objective are as follows:

- The accuracy of the machine learning algorithm (using random forest pixel-based classification) was improved from 89.9% to 92.08% when using an artificial neural network covering a 10x larger area of the same urban environment.
- For a road corridor of length 54.5 km, the accuracy obtained for the white markings of one exit and entry ramp is 91.11 km.
- Using a single class (representing a specific desired feature) allowed for extracting relevant information without redundant data.
- While the training time for deep learning was 5 longer than for traditional machine learning, implementing the trained model over large areas took only minutes rather than hours.

Table 6.6 Summary of results

Corridor	**Accuracy (%)**	**Area/length**
Urban Environment	92.08	22.9 km^2
Road Environment	91.11	54.4 km

6.5 Objective 5: Summary of results and comparison

The table below summarizes and compares the overall objectives to reflect the results.

Table 6.7 Summary of Results

Objective 2: Comparison of primary and integrated Dataset		
	Primary Dataset	**Integrated Dataset**
Overall Accuracy	0.835	0.898
Objective 3: Comparison of pixel-based vs. Object-based image classification		
	Pixel-based	**Object-based**
Overall Accuracy	0.898	0.924
Objective 4: Comparison of Pixel-based Machine Learning vs. Deep Learning		
	Machine learning	**Deep learning**
Area(km^2)	2.1	22
Training Time(hrs)	1	5.5
Classification time(hrs)	2	0.15
Overall Accuracy	0.898	92.08

These results demonstrate the effectiveness of using integrated datasets, object-based image classification, and deep learning for remote sensing applications.

7 Chapter: Prospects

This study demonstrates that machine learning and deep learning techniques offer promising solutions for analyzing multilayer imagery in remote sensing. With further development, these methods have the potential to significantly enhance the accuracy and efficiency of image analysis, particularly in areas such as land use classification, environmental monitoring, and disaster management.

7.1 Multispectral & Hyperspectral Imagery

Deep learning is a powerful technique for extracting useful features and patterns from complex and high-dimensional data, such as multispectral and hyperspectral imagery (MSI and HSI). MSI and HSI are widely used in remote sensing, medical diagnosis, food inspection, and other fields, as they can capture the spectral signature of a target with high spatial and spectral resolution. However, MSI and HSI also pose challenges for traditional image processing and analysis methods, such as noise, redundancy, dimensionality, and variability.

Deep learning classifiers can overcome these challenges by learning hierarchical representations of the data, which can enhance the models' discriminative ability and generalization performance. In recent years, various deep learning classifiers have been proposed for MSI and HSI classification, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs), generative adversarial networks (GANs), and neural graph networks (GNNs). These classifiers can exploit the spatial-spectral correlation of the data, leverage prior knowledge or auxiliary information, and handle imbalanced or scarce labeled data. A comprehensive review of deep learning classifiers for MSI and HSI can be found in (Paoletti et al., 2019). Some examples of state-of-the-art deep-learning-empowered computational spectral imaging methods are discussed (Huang et al., 2022).

7.2 InSAR

Interferometric Synthetic Aperture Radar (InSAR) is a powerful remote sensing technique that can measure surface deformation and topography by exploiting the phase difference between two or more Synthetic Aperture Radar (SAR) images. However, inSAR data processing is often affected by various sources of noise and errors, such as atmospheric effects, scattering mechanisms, acquisition geometry, and temporal decorrelation. Therefore, accurate inSAR phase filtering and coherence estimation are essential steps to improve the quality and reliability of the final results(Murdaca et al., 2022; X. Sun et al., 2020).

In recent years, deep learning has emerged as a promising approach to address the challenges of inSAR data processing. Deep learning is a branch of machine learning that uses artificial neural networks to learn complex and nonlinear patterns from large amounts of data. Deep learning can offer several advantages over traditional methods, such as adaptability, scalability, automation, and generalization. Several studies have proposed deep learning frameworks for inSAR phase filtering and coherence estimation using different network architectures, loss functions, and training strategies.(Aguiar et al., 2022; X. Sun et al., 2020).

These frameworks can perform better than classical and non-deep learning-based methods in accuracy, efficiency, and robustness. Moreover, deep learning can also enable new applications of inSAR data, such as target classification, change detection, and anomaly detection.

7.3 Temporal Analysis and Change Detection:

Another potential application of the deep learning techniques and integrated data demonstrated in this research project is the identification of distress phenomena, such as road cracks, using satellite images. By taking periodic images of the same area, the growth and severity of damages could be compared over time using machine learning algorithms hence the change detection. This process is largely automatic, requires minimal human intervention, and can handle large amounts of data, making it well-suited to today's available data sets. Additionally, because deep learning results are constant if the parameters are the same, comparing different results is always accurate and reliable without the potential for human error. The information obtained through this process could then be used to develop a solution for end customers, such as a pavement maintenance plan, based on the identified distress phenomena and relevant pavement repair guidelines.

However, future research still needs to address some limitations and open issues. For example, how to effectively exploit the temporal information of multitemporal images to capture the dynamic changes over time; how to design more robust and adaptive deep learning models that can handle different types of changes and different scenarios; how to improve the interpretability and explain the ability of deep learning models for change detection; and how to integrate deep learning with other techniques, such as object-based image analysis, transfer learning, and domain adaptation, to enhance the performance and generalization of change detection. These are some of the promising directions that can advance the field of deep learning and temporal analysis with change detection.

- (Khelifi & Mignotte, 2020)
- (Khoshboresh-Masouleh & Shah-Hosseini, 2021)

7.4 Natural Hazard Risk Management:

Natural hazards are events of natural origin that can cause disruption and devastation to society, nature, and beyond. Examples of natural hazards include floods, earthquakes, landslides, wildfires, and volcanic eruptions. Managing natural hazards involves understanding their causes, impacts, and probabilities and developing strategies to reduce risks and enhance resilience. However, natural hazard risk management faces many challenges, such as data scarcity, uncertainty, complexity, and dynamicity.

One of the emerging technologies that can potentially enhance natural hazard risk management is artificial intelligence (AI), which refers to the ability of machines to perform tasks that normally require human intelligence. AI can help process large amounts of data, extract useful information, learn from patterns, and make predictions and decisions. Among the various branches of AI, deep learning is a subfield that uses neural networks to learn from data hierarchically and nonlinearly. Deep learning has shown remarkable success in many domains, such as computer vision, natural language processing, speech recognition, and recommender systems.

In recent years, deep learning has also been applied to various aspects of natural hazard risk management, such as data collection, hazard detection, impact assessment, and emergency response. For example, deep learning can help analyze satellite imagery to monitor natural hazards and their impacts (Kuglitsch et al., 2022; J. Sun et al., 2022) or use sensor networks to detect and forecast flash floods and avalanches (Kuglitsch et al., 2022; *NHESS – Special Issue – Advances in Machine Learning for Natural Hazards Risk Assessment*, n.d.) Deep learning can also help recognize natural hazard entities from text data, such as names of hazards, locations, dates, and damages. Furthermore, deep learning can help optimize evacuation routes, generate natural language summaries, or facilitate stakeholder communication and collaboration.

However, deep learning also has some limitations and challenges that must be addressed before it can be widely adopted for natural hazard risk management. These challenges include data quality and availability, interpretability and explainability, ethical and social implications, and integration with other methods and models. Therefore, it is essential to foster interdisciplinary, multistakeholder, and international collaboration to develop standards and best practices that facilitate the implementation of deep learning for natural hazard risk management. By doing so, one can harness the potential of deep learning to improve the ability to manage natural hazards and reduce risks.

7.5 Smart irrigation management

Smart irrigation management is a crucial aspect of sustainable agriculture, as it aims to optimize water use and crop yield while minimizing environmental impacts. One of the challenges of smart irrigation management is to accurately estimate the soil moisture and evapotranspiration (ET) of the field, which are influenced by various factors such as weather, soil type, crop type, and irrigation schedule. Traditional methods of measuring soil moisture and ET rely on physical sensors that are costly, labor-intensive, and prone to errors. Therefore, there is a need for alternative methods that can provide reliable and timely information for irrigation decision-making.

One promising method is deep learning, a branch of machine learning that can learn complex patterns and relationships from large amounts of data. Deep learning can be applied to model soil moisture and ET sensor readings based on weather data inputs, such as temperature, humidity, precipitation, and solar radiation. For example,(Abioye et al., 2022) reviewed the research trend and applicability of different machine learning techniques for precision irrigation management and discussed how digital farming solutions, such as mobile and web frameworks, can enable the deployment of developed machine learning models for use by farmers.

(Goap et al., 2018) presented an open-source technology-based smart system to predict the irrigation requirements of a field using the sensing of ground parameters like soil moisture, soil temperature, and environmental conditions along with the weather forecast data from the Internet.

Another example of using deep learning for smart irrigation management is using long short-term memory (LSTM), a recurrent neural network that can capture temporal dependencies and learn from sequential data. LSTM can be used to model the sensor readings of soil moisture, and ET based on the historical data of the field and provide predictions for future irrigation needs. For instance, (Sami et al., 2022) developed an LSTM-based smart system that offers readings for irrigation based on the predictive analysis of temperature, soil moisture, and humidity. They also validated their approach using physical and neural sensor readings to avoid malfunctions. LSTM-based models can offer advantages such as high accuracy, low computational cost, and adaptability to changing conditions.

7.6 Open-Source Contribution:

In addition to this practical application, the study's results could also be used to contribute to the broader field of mapping and GIS by developing an open-source platform, such as a plugin for QGIS, to support the use of deep learning techniques. This would not only make it easier for others to utilize the powerful capabilities of deep learning in their work, but it could also help advance the field by making these techniques more accessible and widely adopted.

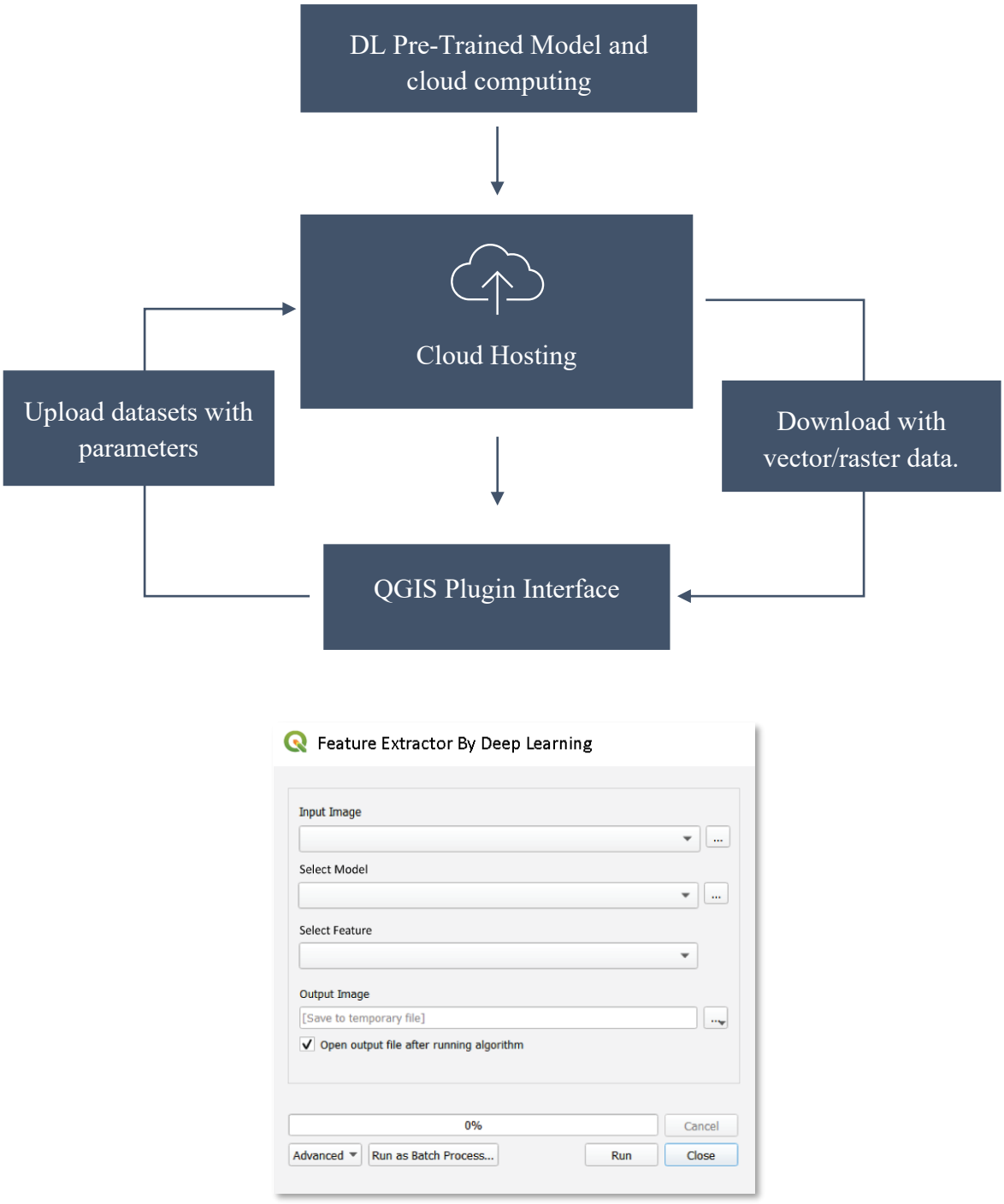


Figure 7.1 On-going plugin development

In addition to these prospects, there are several other potential areas of future development for machine learning and deep learning in remote sensing, including:

- *Object detection and tracking*: In remote sensing imagery, machine learning and deep learning techniques can detect and track specific objects, such as vehicles or buildings. This can be particularly useful for urban planning and traffic monitoring applications.
- *Cloud computing and parallel processing*: As the volume and complexity of remote sensing data continue to grow, cloud computing and parallel processing may become increasingly important for efficiently analyzing such data. Machine learning and deep learning algorithms can be adapted to take advantage of these technologies, allowing for faster and more scalable analysis of remote sensing imagery.
- *Automated feature extraction:* By applying machine learning and deep learning techniques to multilayer imagery, it may be possible to automatically extract features such as roads, buildings, and vegetation without manual annotation. This can significantly reduce the time and cost required for image analysis.
- *Transfer learning:* Transfer learning is a technique that allows machine learning models to be trained on one dataset and then applied to another related dataset. In remote sensing, transfer learning could be used to adapt existing models to new environments, such as different geographic regions or different types of imagery.

7.7 Conclusion:

In this research project, using satellite imagery, machine learning was used to identify different features in an urban corridor. Objective 2 compared the primary dataset (Dataset 1) to the integrated dataset (Dataset 2), which included elevation data. The results showed that the integrated dataset improved classification performance and object-based Classification was superior to pixel-based Classification in an urban environment. Objective 4 focused on using deep learning, specifically convolutional neural networks, for corridor mapping in real-life applications. This method effectively identified specific features in large areas, such as white markings on roads, and required less human intervention than traditional machine learning methods. Overall, it was concluded that deep learning techniques and integrated data could be useful for real-life corridor mapping applications.

8 References

Abioye, E. A., Hensel, O., Esau, T. J., Elijah, O., Abidin, M. S. Z., Ayobami, A. S., Yerima, O., & Nasirahmadi, A. (2022). Precision Irrigation Management Using Machine Learning and Digital Farming Solutions. *AgriEngineering 2022, Vol. 4, Pages 70-103*, *4*(1), 70–103. https://doi.org/10.3390/AGRIENGINEERING4010006

Aguiar, P., Cunha, A., Bakon, M., Ruiz-Armenteros, A. M., & Sousa, J. J. (2022). PS-InSAR Target Classification Using Deep Learning. *International Geoscience and Remote Sensing Symposium (IGARSS)*, *2022-July*, 2931–2934. https://doi.org/10.1109/IGARSS46834.2022.9883937

Belgiu, M., & Drăgu, L. (2016). Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing*, *114*, 24–31. https://doi.org/10.1016/J.ISPRSJPRS.2016.01.011

Chaudhary, V., Buttar, P. K., & Sachan, M. K. (2022). Satellite imagery analysis for road segmentation using U-Net architecture. *Journal of Supercomputing*, *78*(10), 12710–12725. https://doi.org/10.1007/S11227-022-04379-6/METRICS

Francini, M., Salvo, C., & Vitale, A. (2023). Combining Deep Learning and Multi-Source GIS Methods to Analyze Urban and Greening Changes. *Sensors 2023, Vol. 23, Page 3805*, *23*(8), 3805. https://doi.org/10.3390/S23083805

Goap, A., Sharma, D., Shukla, A. K., & Rama Krishna, C. (2018). An IoT based smart irrigation management system using Machine learning and open source technologies. *Computers and Electronics in Agriculture*, *155*, 41–49. https://doi.org/10.1016/J.COMPAG.2018.09.040

Huang, L., Luo, R., Liu, X., & Hao, X. (2022). Spectral imaging with deep learning. *Light: Science & Applications 2022 11:1*, *11*(1), 1–19. https://doi.org/10.1038/s41377-022-00743-6

Khelifi, L., & Mignotte, M. (2020). Deep Learning for Change Detection in Remote Sensing Images: Comprehensive Review and Meta-Analysis. *IEEE Access*, *8*, 126385–126400. https://doi.org/10.1109/ACCESS.2020.3008036

Khoshboresh-Masouleh, M., & Shah-Hosseini, R. (2021). *Deep few-shot learning for bi-temporal building change detection*. https://arxiv.org/abs/2108.11262v2

k-means clustering - Wikipedia. (n.d.). Retrieved April 12, 2023, from https://en.wikipedia.org/wiki/K-means_clustering

Kuglitsch, M. M., Pelivan, I., Ceola, S., Menon, M., & Xoplaki, E. (2022). Facilitating adoption of AI in natural disaster management through collaboration. *Nature Communications 2022 13:1*, *13*(1), 1–3. https://doi.org/10.1038/s41467-022-29285-6

Lu, X., Zheng, X., & Yuan, Y. (2017). Remote sensing scene classification by unsupervised representation learning. *IEEE Transactions on Geoscience and Remote Sensing*, *55*(9), 5148–5157. https://doi.org/10.1109/TGRS.2017.2702596

Ma, L., Liu, Y., Zhang, X., Ye, Y., Yin, G., & Johnson, B. A. (2019). Deep learning in remote sensing applications: A meta-analysis and review. *ISPRS Journal of Photogrammetry and Remote Sensing*, *152*, 166–177. https://doi.org/10.1016/J.ISPRSJPRS.2019.04.015

Murdaca, G., Rucci, A., & Prati, C. (2022). Deep Learning for InSAR Phase Filtering: An Optimized Framework for Phase Unwrapping. *Remote Sensing 2022, Vol. 14, Page 4956*, *14*(19), 4956. https://doi.org/10.3390/RS14194956

Myint, S. W., Gober, P., Brazel, A., Grossman-Clarke, S., & Weng, Q. (2011). Per-pixel vs. object-based classification of urban land cover extraction using high spatial resolution imagery. *Remote Sensing of Environment*, *115*(5), 1145–1161. https://doi.org/10.1016/J.RSE.2010.12.017

NHESS – Special issue – Advances in machine learning for natural hazards risk assessment. (n.d.). Retrieved April 12, 2023, from https://nhess.copernicus.org/articles/special_issue1189.html

Paoletti, M. E., Haut, J. M., Plaza, J., & Plaza, A. (2019). Deep learning classifiers for hyperspectral imaging: A review. *ISPRS Journal of Photogrammetry and Remote Sensing*, *158*, 279–317. https://doi.org/10.1016/J.ISPRSJPRS.2019.09.006

Prakash, N., Manconi, A., & Loew, S. (2020). Mapping Landslides on EO Data: Performance of Deep Learning Models vs. Traditional Machine Learning Models. *Remote Sensing 2020, Vol. 12, Page 346*, *12*(3), 346. https://doi.org/10.3390/RS12030346

Random Forest Algorithms - Comprehensive Guide With Examples. (n.d.). Retrieved April 12, 2023, from https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/

randomforest2001. (n.d.).

Sami, M., Khan, S. Q., Khurram, M., Farooq, M. U., Anjum, R., Aziz, S., Qureshi, R., & Sadak, F. (2022). A Deep Learning-Based Sensor Modeling for Smart Irrigation System. *Agronomy 2022, Vol. 12, Page 212*, *12*(1), 212. https://doi.org/10.3390/AGRONOMY12010212

Satellite Image Segmentation: a Workflow with U-Net | by Vooban | vooban AI | Medium. (n.d.). Retrieved April 12, 2023, from https://medium.com/vooban-ai/satellite-image-segmentation-a-workflow-with-u-net-7ff992b2a56e

Sun, J., Liu, Y., Cui, J., & He, H. (2022). Deep learning-based methods for natural hazard named entity recognition. *Scientific Reports 2022 12:1*, *12*(1), 1–15. https://doi.org/10.1038/s41598-022-08667-2

Sun, X., Zimmer, A., Mukherjee, S., Kottayil, N. K., Ghuman, P., & Cheng, I. (2020). DeepInSAR—A Deep Learning Framework for SAR Interferometric Phase Restoration and Coherence Estimation. *Remote Sensing 2020, Vol. 12, Page 2340*, *12*(14), 2340. https://doi.org/10.3390/RS12142340

Thanh Noi, P., & Kappas, M. (2017). Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery. *Sensors 2018, Vol. 18, Page 18*, *18*(1), 18. https://doi.org/10.3390/S18010018

U-Net Explained | Papers With Code. (n.d.). Retrieved April 12, 2023, from https://paperswithcode.com/method/u-net

U-Net for Semantic Segmentation on Unbalanced Aerial Imagery | by Amirhossein Heydarian | Towards Data Science. (n.d.). Retrieved April 12, 2023, from https://towardsdatascience.com/u-net-for-semantic-segmentation-on-unbalanced-aerial-imagery-3474fa1d3e56

U-Nets with ResNet Encoders and cross connections | by Christopher Thomas BSc Hons. MIAP | Towards Data Science. (n.d.). Retrieved April 12, 2023, from https://towardsdatascience.com/u-nets-with-resnet-encoders-and-cross-connections-d8ba94125a2c

Weng, W., & Zhu, X. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *IEEE Access*, *9*, 16591–16603. https://doi.org/10.1109/ACCESS.2021.3053408

What is Random Forest? | IBM. (n.d.). Retrieved April 12, 2023, from https://www.ibm.com/topics/random-forest

Zhang, T., Su, J., Xu, Z., Luo, Y., & Li, J. (2021). Sentinel-2 Satellite Imagery for Urban Land Cover Classification by Optimized Random Forest Classifier. *Applied Sciences 2021, Vol. 11, Page 543*, *11*(2), 543. https://doi.org/10.3390/APP11020543

Zhang, X., Han, L., Han, L., & Zhu, L. (2020). How Well Do Deep Learning-Based Methods for Land Cover Classification and Object Detection Perform on High Resolution Remote Sensing Imagery? *Remote Sensing 2020, Vol. 12, Page 417*, *12*(3), 417. https://doi.org/10.3390/RS12030417