University of Novi Sad
Faculty of Sciences
Department of Mathematics and Informatics

Student:
Sonja Jamina

Assessing U-Net-Based Semantic Segmentation Models for Vineyard Row Detection in UAV Imagery

Master Thesis

Mentor:
Dr Nikola Obrenović

Novi Sad, 2026

I would like to sincerely thank my professors, Dr Nikola Obrenović and Dr Oskar Marko, and assistant, Dimitrije Stefanović, for their guidance, support, and encouragement throughout this work.

I'm also deeply grateful to my family and friends for their continuous support and motivation. Your belief in me made this journey possible.

Abstract

This thesis investigates the automatic segmentation of vineyard rows from UAV-acquired multispectral orthomosaic imagery, as a basis for inferring a drivable path for an unmanned ground vehicle (UGV) operating between the rows - a step toward bringing autonomous navigation into a centuries-old agricultural tradition. Two deep learning segmentation architectures, U-Net and Dense U-Net, were trained and evaluated across four spectral channel combinations: RGB, RGB-NIR, RGB-RE, and Green-RE-NIR, using manually digitized vine-row polygons as ground truth. Model performance was assessed using Intersection over Union, Dice coefficient, pixel accuracy, precision, recall, F1-score, and specificity. The availability of a near-infrared or red-edge spectral band was found to be the dominant factor in segmentation quality: all three such combinations achieved Dice coefficients above 0.87, while the purely visible-spectrum RGB combination performed measurably worse, and U-Net matched or outperformed the more complex Dense U-Net across every tested configuration. These findings provide a concrete, data-driven basis for selecting sensor and architecture configurations when deploying vineyard row segmentation as part of a real-world UGV navigation system, helping precision viticulture advance without losing sight of the tradition it is built on.

Contents

Abstract **2**
List of Figures and Tables **4**
1. Introduction **5**
1.1 Outline 6
2. Related work **7**
3. Materials and methods **10**
3.1 Study Site and UAV Data Acquisition 10
3.2 Data Annotation and Ground Truth Generation 10
3.3 Spectral Characteristics of UAV Imagery 12
3.4 Data Processing 14
3.5 Algorithms 15
3.5.1 Convolutional Neural Networks (CNNs) 16
3.5.1.1 Pooling 17
3.5.2 Fully Convolutional Networks (FCNs) 19
3.5.3 Encoder-Decoder Architectures 20
3.5.3.1 Skip Connections 22
3.5.3.2 Atrous (Dilated) Convolutions 22
3.5.3.3 Attention Mechanisms 23
3.5.4 U-Net 24
3.5.5 Dense U-Net 25
3.5.6 Dropout 26
3.5.7 Optimization 26
3.5.8 Batch Normalization 27
3.6 Evaluation Metrics 27
3.6.1 Intersection over Union (IoU) 27
3.6.2 Dice Coefficient 28
3.6.3 Pixel Accuracy 28
3.6.4 Precision 29
3.6.5 Recall 29
3.6.6 F1-Score 29
3.6.7 Specificity 30
4. Experiments and Results **31**
4.1 Experimental Setup 31
4.2 Results and Analysis 32
Effect of spectral channel combination 33
Effect of architecture 34
The RGB + Dense U-Net result 34
Training dynamics and overfitting 34
Qualitative results 35
Per-image variability 36
5. Conclusion and Future Work **38**
Bibliography **40**

List of Figures and Tables

Figure 1: Vineyard Imagery: Ancient Art vs. Modern Drone View

Figure 2: The sun in different wavelengths

Figure 3: Point cloud smart mapping in 3D

Figure 4: Data Annotation process

Figure 5: Spectral band combinations used in this experiment

Figure 6: Data preprocessing and augmentation at three spatial scales

Figure 7: Convolutional neural network pipeline

Figure 8: Illustration of the convolution operation

Figure 9: Types of pooling used in CNN

Figure 10: Schematic illustration of an encoder-decoder architecture

Figure 11: Skip connections in a symmetric encoder-decoder architecture

Figure 12: U-Net Architecture

Figure 13: Dense U-Net Architecture

Figure 14: Test-set Dice coefficient and IoU across all eight trained configurations

Figure 15: Per-epoch training and validation curves

Figure 16: Qualitative segmentation examples

Figure 17: Prediction overlap by simulated flight altitude

Table 1: Comparison between different evaluation metrics

Table 2: Test-set evaluation metrics for all eight trained configurations

Table 3: Test-set metrics for the best-performing configuration, broken down by simulated flight altitude

1. Introduction

Food production is one of the most important aspects of our society, and it has played a crucial role in the survival of our species from the dawn of human civilization. From the simple prehistoric tools to modern-day heavy machinery, every technological advancement was important to ensure sustainability and continue producing sufficient food supplies. Vineyards have been an important part of agriculture and culture since ancient times, with viticulture dating back over 6,000 years. These historical landscapes, celebrated for their contributions to hospitality and trade, have evolved into a blend of tradition and science. Beyond their pivotal role in winemaking, grapes derived from these vineyards are also known for their health benefits, packed with antioxidants like flavonoids and resveratrol. These compounds are celebrated for their ability to neutralize harmful free radicals, potentially reducing the risk of various chronic diseases. Therefore, the cultivation of grapes extends far beyond winemaking. It is also a nutritious food source that plays a key role in maintaining our health. This highlights the vineyards' significance, where they stand not just as the foundation of agriculture but also as sources of valuable nutrients essential for human well-being.

The growing global population is increasing the demand for food, while at the same time, agricultural labor is becoming harder to find, especially as farming expands to remote, less accessible areas far from urban centers.

Now, in the era where technology meets tradition, advancements like Unmanned Aerial Vehicle (UAV) drone imagery and U-Net [1] segmentation are modernizing vineyard management. By embracing these innovations, we honor the deep-rooted significance of vineyards and ensure their enduring legacy in agriculture.

Among these advancements, developing autonomous navigation for agricultural vehicles using high-resolution aerial imagery stands out as a crucial innovation. This Master's thesis focuses on the precise detection of vineyard rows through state-of-the-art image segmentation techniques. Specifically, in this thesis, we will explore the capabilities of UAVs, which are drones controlled remotely or autonomously, and the U-Net deep learning architecture, which is a type of advanced neural network. Combined UAVs and deep learning aim to help in guiding Unmanned Ground Vehicles (UGVs), which are robotic vehicles that navigate autonomously on the ground, through the challenging terrain of vineyards. Combining drones with deep learning allows for efficient and immediate monitoring of vineyards, leading to faster responses to changing conditions.



Figure 1: Vineyard Imagery: Ancient Art [2] vs. Modern Drone View [3]

Figure 1 contrasts this traditional, intensely manual heritage of viticulture with the modern, technology-assisted approach explored in this thesis.

This thesis presents a methodology for the segmentation of vineyard rows using a customized U-Net architecture. The U-Net model, originally designed for biomedical image segmentation, is known for its efficacy in distinguishing between patterns and textures in images. By training this model with UAV-derived images of vineyards, we aim to produce precise segmentation masks that highlight the vine rows, effectively distinguishing them from the surrounding soil and vegetation. We will also consider the varying heights at which drones operate when capturing images, ensuring these methods are adaptable to different altitudes for flexible and thorough data collection.

This thesis investigates how different UAV spectral channel combinations and segmentation architectures affect the accuracy of vineyard row detection, with the goal of establishing a reliable basis for inferring a navigable path for a UGV operating between the rows.

We are moving towards a future where robots and farming work better together, making things more efficient and kind to the environment. In short, this thesis combines high-tech drone images and smart learning to help make farming robots. By advancing precision agriculture, this research contributes to sustainable viticulture, preserving resources, and improving crop quality.

1.1 Outline

This thesis is organized as follows. Chapter 2 covers the related work in the field. Chapter 3 introduces the problem and methodology in more detail. Chapter 4 gives experimental results and discussion, and finally, the thesis is concluded in Chapter 5.

2. Related work

The field of precision agriculture is rapidly advancing through the integration of remote sensing [4] and deep learning [5]. Recent developments have enabled increasingly sophisticated applications, including image segmentation [6], three-dimensional point cloud analysis [7], and intelligent crop management systems [8]. Together, these technologies allow large volumes of agricultural data to be processed into actionable information that supports efficient farm management and informed decision making.

At the heart of these developments are multispectral (MS) sensors [9], which, when mounted on drones or satellites, capture images across multiple wavelengths beyond the visible spectrum. Such imagery supports numerous precision agriculture applications, including crop health monitoring, disease detection, water stress assessment, and vineyard management [10].

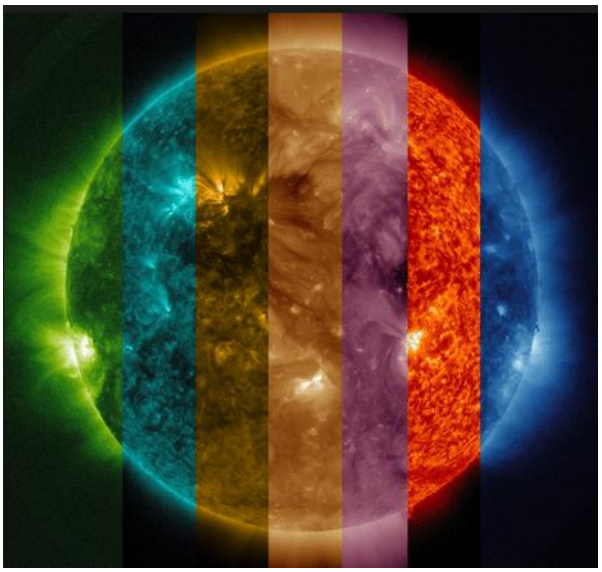


Figure 2: The sun in different wavelengths [12]

Figure 2 illustrates how the same scene appears markedly different when captured across different wavelength bands, underlying the motivation for using multispectral rather than purely visible-light imagery.

Earlier remote sensing applications in agriculture relied primarily on satellite imagery [11]. Although satellite data provide broad spatial coverage, they are often limited by relatively low spatial resolution, infrequent revisit times, and cloud cover. The increasing affordability and accessibility of UAVs have largely overcome these limitations by enabling flexible acquisition of high-resolution imagery tailored to individual fields.

UAV surveys of a vineyard are typically processed into a single orthomosaic: one composite image, corrected for distortion and aligned to real-world coordinates, stitched together from many overlapping aerial photographs to give a seamless, accurate view of the whole field. Precisely detecting which regions of this orthomosaic are vine rows, as opposed to soil or inter-row vegetation, is a necessary step for any further analysis, and reliably obtaining such masks requires the rows to be manually annotated. Such row-level masks also make it possible to map which parts of a vineyard are actually traversable, providing the basis for planning a navigable path for a UGV through the inter-row space.

One of the earliest approaches to vineyard row detection from UAV imagery was proposed by Comba et al. [27]. Their method combines dynamic image segmentation, Hough Space Clustering, and Total Least Squares to identify vineyard rows directly from aerial images while remaining robust to illumination changes, shadows, and background vegetation. The extracted row geometry was further proposed as a basis for autonomous navigation of UGVs within vineyards. Unlike this feature-engineered approach, the method presented in this thesis employs deep learning-based semantic segmentation to identify vineyard rows directly at the pixel level without relying on manually designed geometric features.

Beyond conventional two-dimensional imagery, UAV photogrammetry also enables the reconstruction of dense three-dimensional point clouds. Biglia et al. [24] proposed a density-based clustering algorithm that automatically detects and localises vineyard rows directly from such 3D point clouds without requiring colour or spectral information, reporting localisation accuracy consistent with manual reference measurements. While their approach relies exclusively on the geometric structure of a reconstructed point cloud, the methodology presented in this thesis instead exploits the spectral information contained in multispectral UAV imagery, avoiding the additional computational cost associated with dense 3D reconstruction. Figure 3 illustrates an example of such a 3D point cloud representation of a surveyed area.

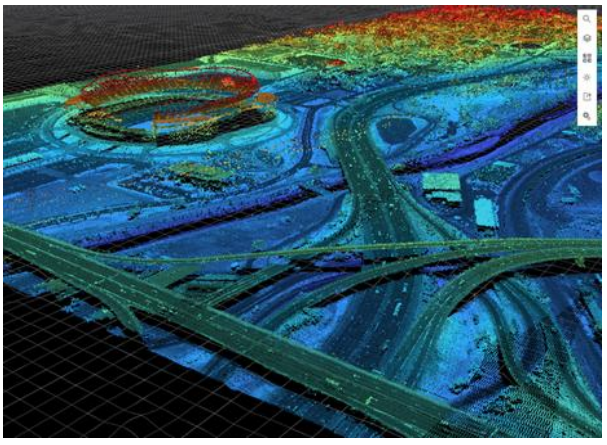


Figure 3: Point cloud smart mapping in 3D [15]

Machine learning has significantly expanded the capabilities of precision agriculture by enabling automated interpretation of remotely sensed imagery. By learning complex spatial and spectral patterns directly from data, modern algorithms support applications including crop monitoring, disease detection, yield estimation, and autonomous agricultural navigation.

As highlighted by Minaee et al. (2022) [6], semantic segmentation architectures such as U-Net and SegNet have become foundational models because of their ability to learn hierarchical image features while preserving fine spatial detail, making them particularly effective for analysing complex outdoor scenes.

Chang et al. [13] proposed an improved DeepLabV3+ architecture - a convolutional encoder-decoder network for semantic segmentation that uses atrous (dilated) convolutions to capture multi-scale spatial context without reducing feature-map resolution (see Section 3.5.3.2), for crop-type identification from high-resolution remote sensing imagery. Their model combines an encoder-decoder framework with an attention mechanism and a Multiscale Feature Enhancement Module (MFEM), improving segmentation performance while maintaining computational efficiency across crops of different scales.

The transition from handcrafted image-processing techniques to deep learning represents a significant evolution in agricultural computer vision. Rather than relying on manually designed geometric or spectral features, deep neural networks learn discriminative representations directly from data, making them more robust to the variability encountered in real-world agricultural environments.

A separate line of research addresses the cost of producing annotated training data for remote sensing applications. Olaniyi et al. [26] presented a systematic review of generative adversarial networks (GANs) for agricultural image augmentation, highlighting their ability to generate realistic synthetic imagery or paired image-mask datasets that reduce the dependence on manually annotated data. Although such methods have seen only limited application to vineyard row segmentation, they represent a promising direction for reducing the substantial manual effort required to produce pixel-level ground truth annotations, an effort that played a central role in this work (Section 3.2).

Overall, previous research demonstrates that classical computer vision, geometric reconstruction, multispectral remote sensing, and deep learning-based image segmentation provide complementary approaches for agricultural scene understanding. Building upon these developments, this thesis investigates semantic segmentation of vineyard rows directly from multispectral UAV imagery to support the subsequent navigation of unmanned ground vehicles.

3. Materials and methods

3.1 Study Site and UAV Data Acquisition

The dataset used in this study was collected within the FLEXIGROBOTS Horizon 2020 project over a commercial vineyard located in Pontevedra, Galicia, Spain (41.955187° N, −8.795055° W). The vineyard is owned by Bodegas Terras Gauda, S.A. and is situated on terrain with an approximately 8.1% slope. The UAV survey covered a portion of the vineyard, and only the area contained within the surveyed field boundary was considered in the subsequent data preparation and analysis.

The imagery was acquired on 12 July 2022 using a DJI Phantom 4 Multispectral UAV at a flight altitude of 30 m. The onboard multispectral camera captures five spectral bands: Blue (450 nm ± 16 nm), Green (560 nm ± 16 nm), Red (650 nm ± 16 nm), Red Edge (730 nm ± 16 nm), and Near-Infrared (840 nm ± 26 nm), with a spatial resolution of 2.12 MP per band. The acquired imagery was processed into a georeferenced multispectral orthomosaic, which served as the source data for the annotation, patch generation, spectral-channel experiments, and segmentation models described in the following sections.

3.2 Data Annotation and Ground Truth Generation

For a supervised segmentation model to learn to distinguish vine rows from the surrounding soil and vegetation, it must be trained on examples where the correct answer is already known. Producing these examples, commonly referred to as ground truth, is one of the most important steps in building a reliable dataset, since the quality of a model's predictions is ultimately limited by the quality of the labels it learns from.

Vine rows can vary considerably in appearance depending on canopy density, the presence of weeds between rows, and lighting conditions at the time of image acquisition. Relying on a fully automatic method to generate labels under these conditions risks introducing systematic errors that are difficult to notice and even harder to correct once they propagate through the rest of the pipeline. For this reason, the ground truth used in this work was produced through manual annotation, so that every row present in the orthomosaic was identified through direct visual inspection rather than inferred indirectly from pixel intensities.

The annotation process was carried out using a geographic information system software QGIS, which allows spatial features to be drawn directly on top of the georeferenced orthomosaic at full resolution. Two distinct sets of features were digitized for the surveyed plot. The first outlines the overall boundary of the field, marking the area that was actually covered by the UAV survey and should therefore

be considered when generating image patches for training. The second consists of individual polygons traced around every vine row visible in the orthomosaic, each one drawn by hand to closely follow the true shape and extent of the row on the ground.

Once digitization was complete, both sets of polygons were converted into pixel-aligned masks matching the resolution and coordinate system of the orthomosaic. The row polygons determine, for every pixel in the image, whether that location belongs to a vine row or to the surrounding background, and together these decisions form the binary ground truth used throughout the remainder of this work. The field-boundary polygon serves a complementary role: it identifies which parts of the orthomosaic fall outside the actual survey area, allowing such regions to be excluded once the imagery is divided into patches, as described in Section 3.4.

This annotation strategy ensures that the labels used to train and evaluate the segmentation models reflect the true position of the vine rows in the field, rather than an approximation derived after the fact from the spectral properties of the imagery.

Figure 4 illustrates this process on the surveyed orthomosaic: the raw imagery (a), the digitized field boundary (b), the individually traced row polygons (c), and their combination into the final binary mask used as ground truth (d).

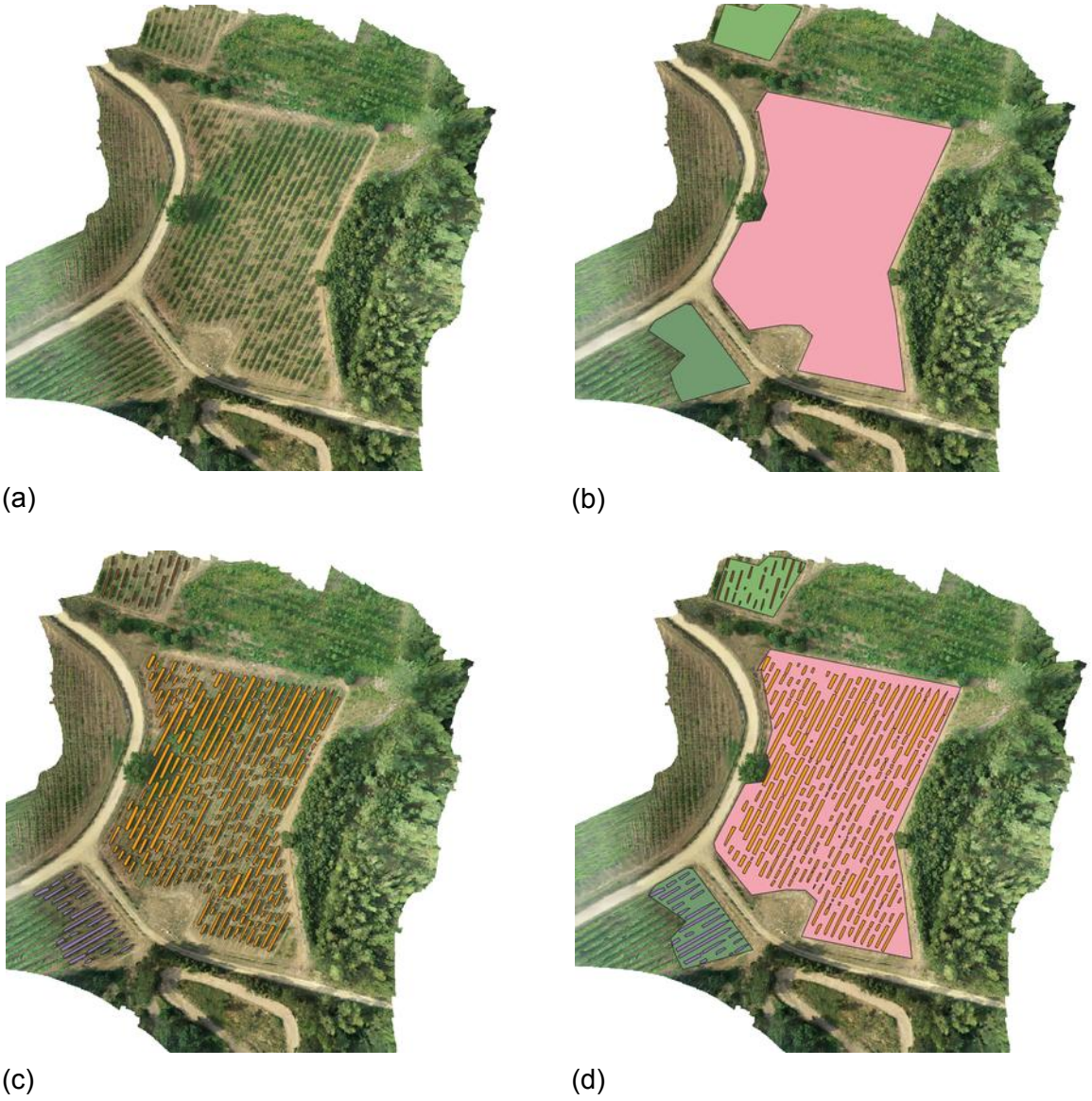


Figure 4: Data annotation process. (a) Raw orthomosaic imagery. (b) Digitized field boundary. (c) Manually traced row polygons. (d) Resulting binary ground-truth mask.

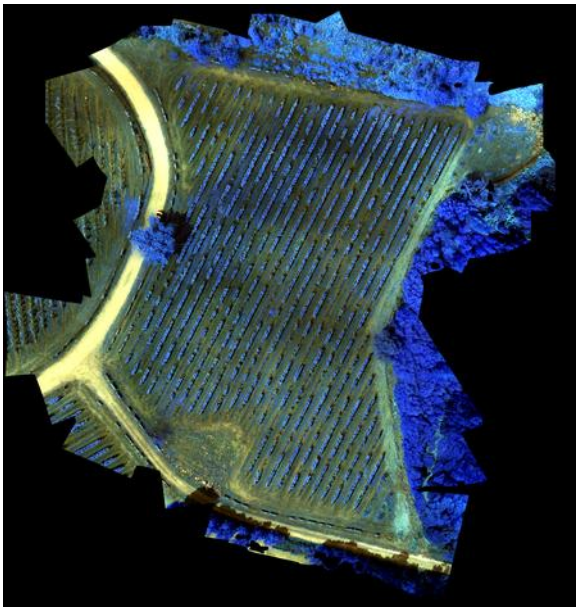
3.3 Spectral Characteristics of UAV Imagery

In remote sensing, analyzing spectral signatures is crucial for extracting valuable information from imagery. The spectral bands, including Red, Green, Blue, Near-Infrared (NIR), and Red Edge, each serve a specific purpose.

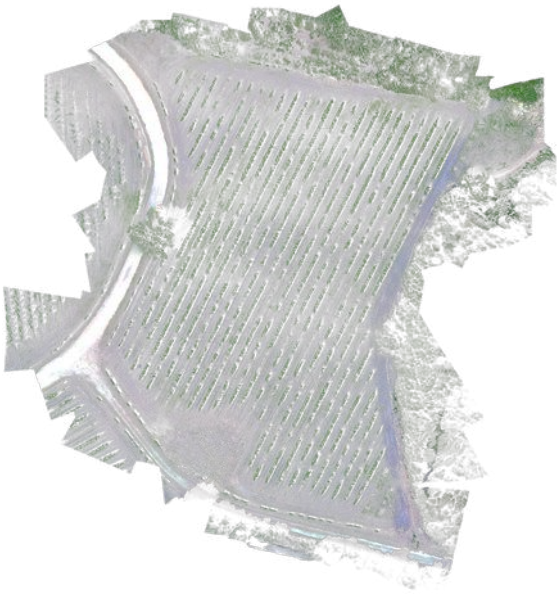
Red is absorbed by vegetation and is useful in soil and geological studies. Green, reflecting off-leaf surfaces, indicates plant health. Blue, ideal for deep water and atmospheric observations, penetrates marine environments. NIR, strongly reflected by healthy vegetation, aids in biomass and vegetation analysis. The Red Edge band, showing plant health status, is applied in precision agriculture and stress detection [16].



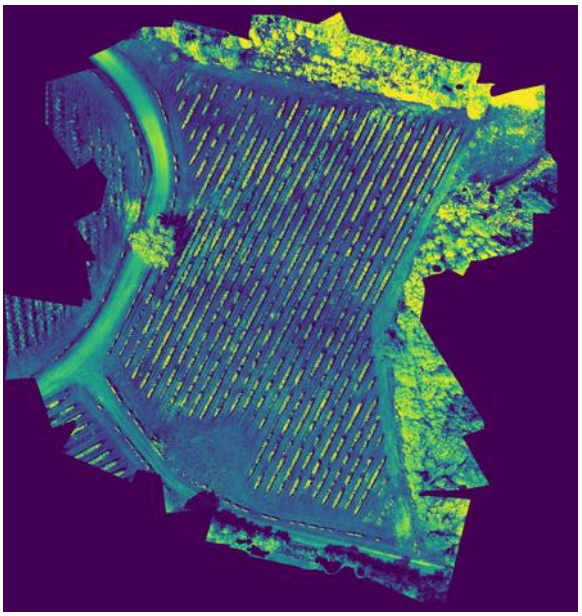
(1)



(2)



(3)



(4)

Figure 5: Spectral band combinations used in this experiment: (1) RGB, (2) Green-RE-NIR, (3) RGB-RE, and (4) RGB-NIR

In the context of remote sensing for this research, the following spectral band combinations are explored:

RGB (Red, Green, Blue): This standard model mimics human vision and is the most common input modality for image segmentation methods more broadly [6].
Green-RE-NIR (Green, Red Edge, NIR): Both the red-edge and green bands are established indicators of chlorophyll content and plant health, and are combined with

NIR in widely used vegetation indices for detailed agricultural and ecological analysis [28].
RGB-RE (Blue, Green, Red, Red Edge): The red-edge band is established as sensitive to chlorophyll absorption and plant stress, making it a valuable addition to RGB for vegetation health detection in environmental monitoring and precision agriculture [28,25].
RGB-NIR (Blue, Green, Red, NIR): Adding NIR to RGB is a well-established basis for vegetation and water-body analysis, as in the widely used NDVI and NDWI indices [29,30,31].

Differences between these spectral bands are illustrated in Figure 5, on an example of a vineyard drone capture.

3.4 Data Processing

The multispectral orthomosaic was divided into image patches that were subsequently used as inputs to the segmentation models. Only patches located within the annotated field boundary were retained, ensuring that regions outside the surveyed area were excluded from the dataset. All model inputs were standardized to a spatial dimension of 512×512 pixels to ensure consistent input dimensions during training and inference. Image processing was performed using OpenCV [17].

To approximate the effect of UAV imagery acquired at different flight altitudes, patches were extracted from the original orthomosaic at three spatial scales: 256×256, 512×512, and 1024×1024 pixels. The original UAV imagery was acquired at a single flight altitude of 30 m; therefore, these scales do not represent additional UAV acquisitions, but simulate changes in the apparent size of vineyard structures by varying the area of the orthomosaic contained within each patch. All patches were subsequently represented as 512×512 model inputs, as illustrated in Figure 6.

The 256×256 source patches cover a smaller area of the orthomosaic and were upsampled to 512×512 pixels using bicubic interpolation (cv.INTER_CUBIC) [18], which estimates each new pixel's value from a weighted combination of the 16 nearest pixels (a 4×4 neighborhood) in the source image, fitted with cubic polynomials. This produces smoother, less blocky results than simpler methods such as nearest-neighbor interpolation. Consequently, vineyard structures appear larger within the final model input, approximating the effect of imagery acquired at a lower flight altitude. The native 512×512 patches were used without spatial resizing and represent the reference scale. In contrast, the 1024×1024 source patches cover a larger area of the orthomosaic and were downsampled to 512×512 pixels using area-based interpolation (cv.INTER_AREA) [19], which computes each output pixel as a weighted average of all source pixels that fall within its corresponding area in the original image. This averaging prevents aliasing artifacts that occur when pixels are simply dropped during shrinking, making it the recommended OpenCV method

for image reduction. Vineyard structures therefore appear smaller in the resulting model input, approximating the effect of imagery acquired at a higher flight altitude. Figure 6 shows examples of the resulting patches at all three scales.

In addition to the scale transformations, data augmentation was applied to increase the size and spatial diversity of the dataset. For each retained patch, three rotated versions (90°, 180°, and 270°) were generated, together with a horizontally flipped version and its corresponding three rotations. This resulted in eight spatial variants of each original patch. The same transformations were applied to the corresponding ground-truth masks, preserving pixel-level alignment between each image and its segmentation label. Figure 6 illustrates the original, flipped, and rotated variants for each of the three spatial scales.

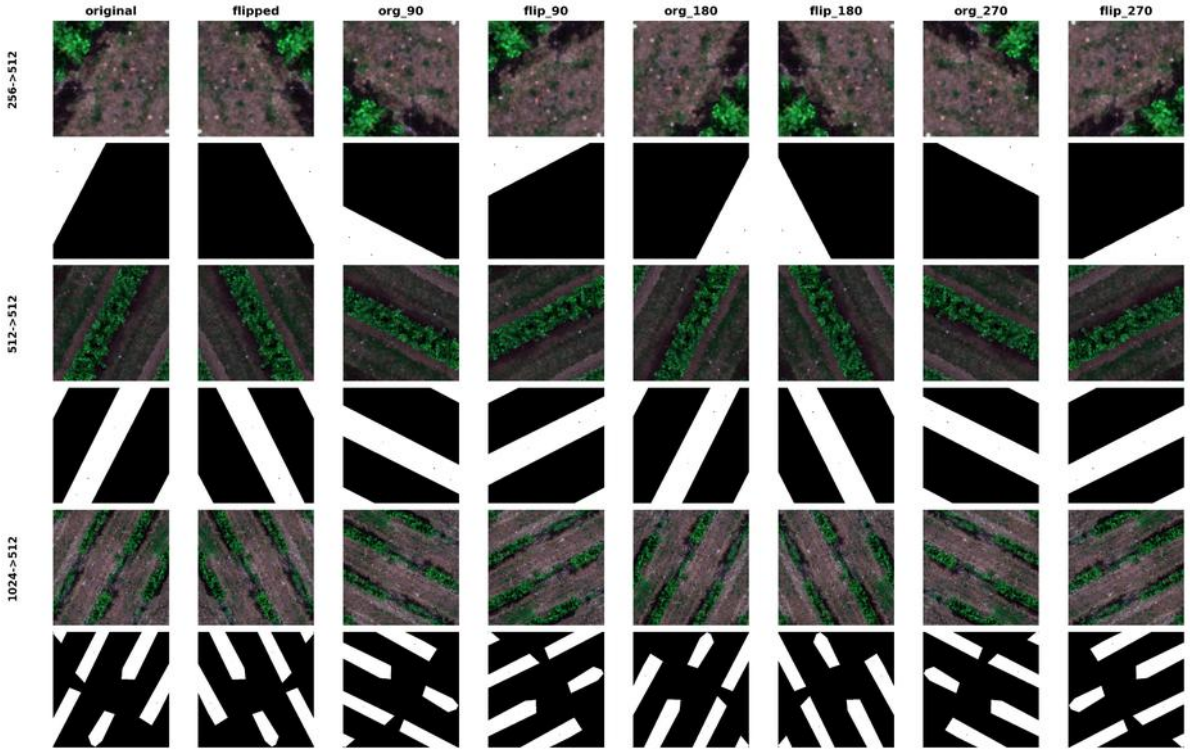


Figure 6: Data preprocessing and augmentation at three spatial scales.
Rows correspond to source patches of 256×256, 512×512, and 1024×1024 pixels, all represented as 512×512 model inputs. The 256×256 patches are upsampled to approximate a lower flight altitude, the native 512×512 patches represent the reference scale, and the 1024×1024 patches are downsampled to approximate a higher flight altitude. Columns show the original patch, its horizontal flip, and the corresponding 90°, 180°, and 270° rotations. Each image is shown together with its corresponding ground-truth segmentation mask.

3.5 Algorithms

This thesis explores the field of semantic segmentation, an important part of computer vision that enhances the interaction between computers and the visual

world. Semantic segmentation involves the process of dividing an image into meaningful parts, each labeled according to the class of the objects they represent. This division helps machines understand and interpret visual data, which is crucial in applications like UAV-based vineyard monitoring.

Within the broader context of Data Science, semantic segmentation is a key application of Computer Vision, particularly through the use of Convolutional Neural Networks (CNNs) [20]. These networks apply layers of filters to images to highlight important features and structures, simplifying the task of segmenting images into meaningful parts. This process enables computers to accurately identify and classify different sections of an image. This overall processing pipeline is illustrated in Figure 7.

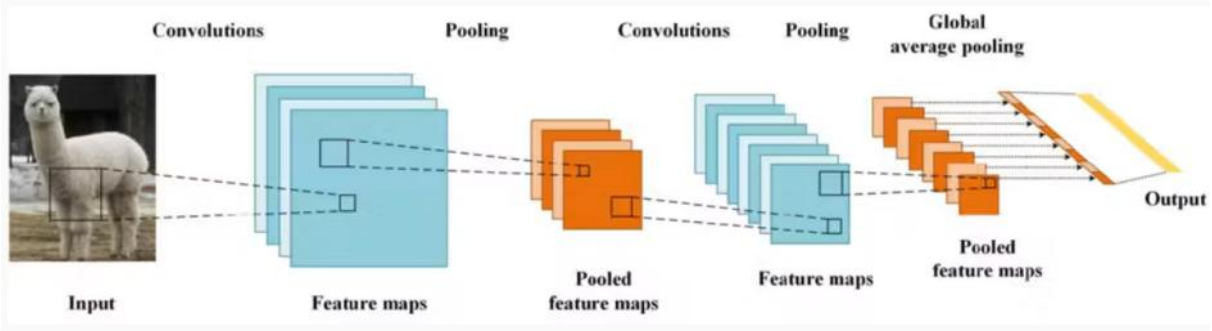


Figure 7: Convolutional neural network pipeline, showing an input image processed through repeated convolution and pooling stages to produce stacked feature maps and a final output [35].

3.5.1 Convolutional Neural Networks (CNNs)

CNNs form the backbone of modern semantic segmentation models. They use convolutional layers to learn hierarchical feature representations from images, starting from low-level features like edges and textures to high-level features such as shapes and patterns. This feature extraction capability is critical for identifying and separating objects in an image.

A convolutional layer applies a kernel - a small matrix of learnable weights, typically much smaller than the input (e.g., 3×3) - to the input image. The kernel slides across every position of the input. At each position, it computes a weighted sum of the pixel values it currently overlaps, producing one value of the output feature map. This sliding-window operation is illustrated in Figure 8. Formally, a square kernel of size k, applied over a matrix X representing the original image, takes the following transformation to produce the latent representation Y:

$$Y(i,j) = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} w(m,n) X(i+m, j+n) + b$$

Where:

- $Y(i,j)$ is the output feature at the position (i,j).
- $X(i+m,\ j+n)$ is the input value at position $(i+m,\ j+n)$.
- $w(m,n)$ is the weight of the kernel at position (m,n).
- b is the bias term.
- k defines the full size of the kernel (e.g., k=3 for a 3×3 kernel), so m and n each range over the k positions 0 to k−1.

CNNs process the image through multiple layers, with deeper layers capturing high-level features like shapes and objects, while earlier layers capture low-level features like edges and textures.

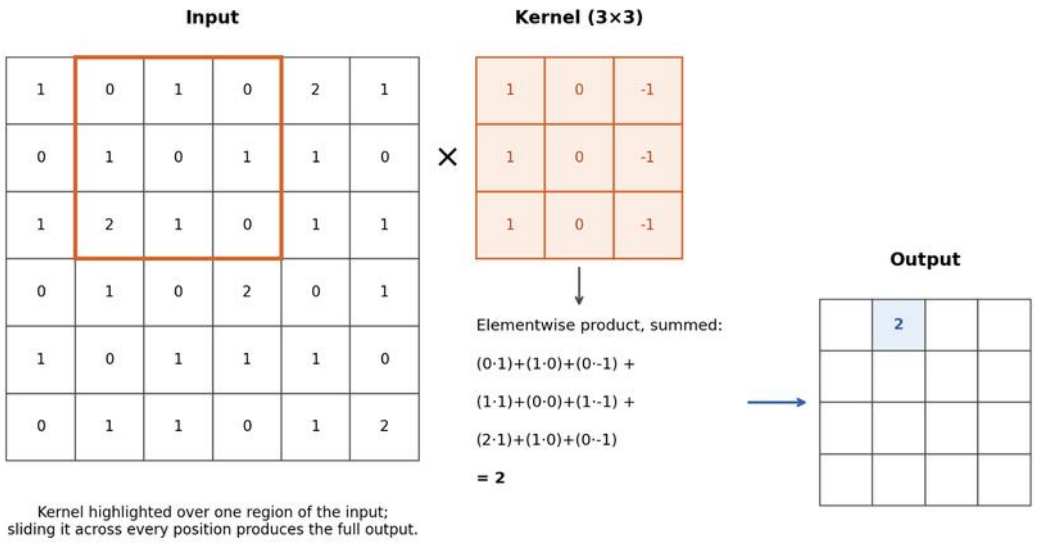


Figure 8: Illustration of the convolution operation. A 3×3 kernel is positioned over one 3×3 region of the input (highlighted), and the elementwise product between the kernel and that region is summed to produce a single value in the corresponding position of the output feature map. Sliding the kernel across every position of the input produces the complete output.

3.5.1.1 Pooling

Pooling is a fundamental operation in CNNs that reduces the spatial dimensions of feature maps while retaining essential information. It helps in:

- Dimensionality Reduction: Reducing the number of computations and parameters.
- Translation Invariance: Ensuring robustness to small shifts and distortions in the input image.

There are two main types of pooling used in segmentation tasks:

1. Max Pooling:

   Extracts the maximum value from a sliding window across the input feature map.

   $$Y(i,j) = max_{(m,n)\in R(i,j)} X(m,n)$$

   Where:

   - $X(m,n)$ is the input feature at position (m,n).
   - $R(i,j)$ is the receptive field (e.g., 2×2) for position (i,j).
   - $Y(i,j)$ is the pooled output at (i,j).

2. Average Pooling:

   Computes the average value of the features within a sliding window.

   $$Y(i,j) = \frac{1}{|R(i,j)|} \sum_{(m,n)\in R(i,j)} X(m,n)$$

   Where:

   - $X(m,n)$ is the input feature at position (m,n).
   - $|R(i,j)|$ is the number of elements in the receptive field for position (i,j).
   - $Y(i,j)$ is the pooled output at (i,j)

Figure 9 illustrates pooling and the differences between the two approaches.

Key Parameters in Pooling:

- Kernel Size (k×k):

  Defines the dimensions of the pooling window. For example, a 2×2 kernel aggregates information from 4 pixels.

- Stride:

  Determines the step size for the pooling window. A stride of s moves the window by s pixels. Larger strides result in more aggressive dimensionality reduction:

  $$H_{out} = \lfloor \frac{H-k+2p}{s} + 1 \rfloor,\ W_{out} = \lfloor \frac{W-k+2p}{s} + 1 \rfloor$$

  Where:

- H_{out} is the height of the output feature map.
- W_{out} is the width of the output feature map.
- H, W are the height and width of the input feature map.
- k is the size of the pooling kernel (e.g., 2×2).
- p is the padding added to the input feature map.
- s is the stride (step size of the pooling window).
- $\lfloor \cdot \rfloor$ is the floor function, which ensures that the output dimensions are integers

- Padding:

  Adds extra pixels around the border of the input feature map, ensuring that pooling windows cover edge pixels. Padding is critical for maintaining alignment and spatial consistency.

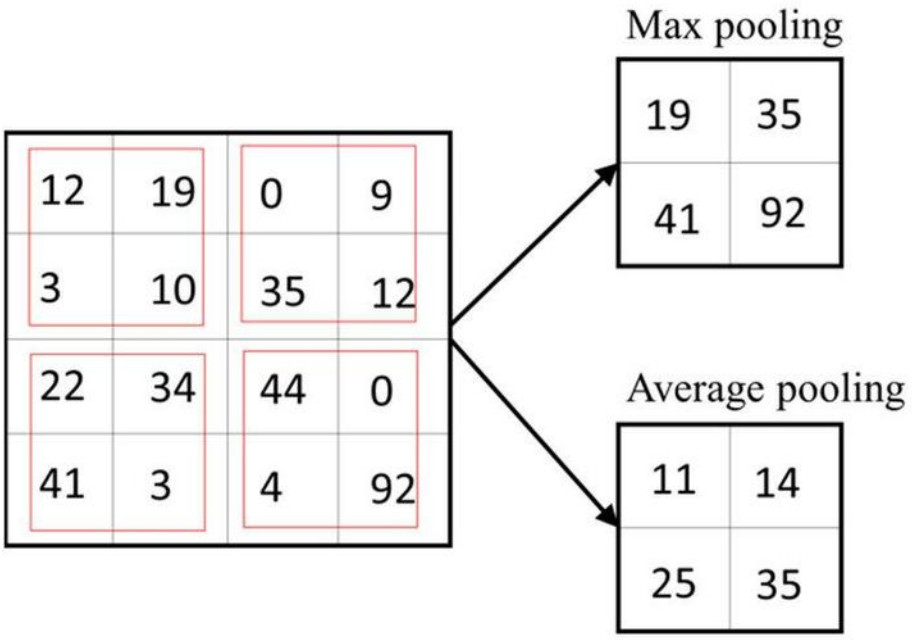


Figure 9: Types of pooling used in CNN.

3.5.2 Fully Convolutional Networks (FCNs)

Traditional CNNs were adapted for semantic segmentation through the development of Fully Convolutional Networks (FCNs) [21]. Unlike standard CNNs that output a single label for the entire image, FCNs replace fully connected layers with convolutional layers, enabling the model to produce a dense pixel-wise prediction map. This innovation laid the groundwork for subsequent architectures tailored for segmentation tasks.

Traditional CNN is fully connected:

$$y = Wx + b$$

where W is a weight matrix, x is the vector of input features, and b is the bias term.

FCNs replace this with a 1×1 convolution:

$$Y(i,j) = w\,X(i,j) + b$$

Where:

- $X(i,j)$ is the input feature at position (i,j).
- w is the weight applied at that position.
- $Y(i,j)$ is the predicted value for each pixel.
- b is the bias term.

FCNs also incorporate upsampling layers (e.g., deconvolution or bilinear interpolation) to reconstruct the original spatial dimensions of the image.

3.5.3 Encoder-Decoder Architectures

Encoder-decoder combination is a widespread architecture for learning latent features for compressing or reconstructing images, which finds applications in all branches of industry.
Many segmentation models, such as U-Net and SegNet, use an encoder-decoder design. The encoder compresses the input image into a compact representation by progressively reducing its spatial dimensions while retaining semantic information.

Encoder formula:

$$x_{l+1} = f(W_l * x_l + b_l)$$

Where:

- x_{l+1} is the input to the l-th layer.
- W_l is the weight matrix for the l-th layer.
- f is the activation function (e.g., ReLU).

The given formula is an example of compressing a simple vector using a neural network, not specifically an image using a convolutional neural network. (In CNN encoder would reduce spatial dimensions using convolution and pooling layers.) However, the fundamental principle remains the same: the weight matrix W_l must have dimensions such that its height (number of rows) is smaller than its width

(number of columns). If this condition isn't met, dimensionality won't decrease in the subsequent step.

Thus, for dimensionality reduction (compression), W_l it should satisfy:

- $W_l \in R^{m \times n}$, $where\, m < n$

This ensures the output vector x_{l+1} has a lower dimensionality compared to the input vector x_l.

The decoder then reconstructs the spatial dimensions to produce a segmentation map, using features extracted by the encoder.

Decoder formula:

$$x_{l+1} = f(W_l^T * x_l + b_l)$$

The given formula illustrates the reconstruction of the original dimensionality using transposed convolutions. Transposed convolution involves "unpooling" the feature maps to increase their resolution.

The decoder produces a segmentation map of the matching dimension, instead of the realistic reconstruction of the input image. To achieve accurate segmentation, the model needs to be trained with a set of labels, i.e., one needs a ground truth segmentation map for each input image, to evaluate the produced outputs.
This general structure is illustrated in Figure 10.

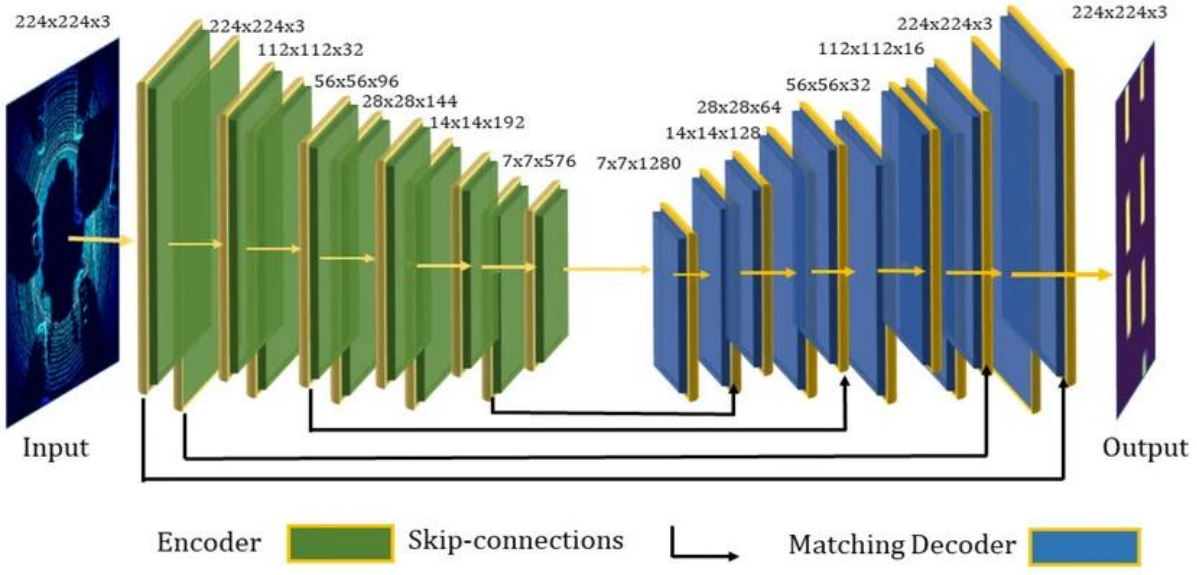


Figure 10: Schematic illustration of an encoder-decoder architecture, where the left-hand side is an input passed through a series of encoder layers, and the right-hand side is the decoder's output feature map, arrows represent skip connections concatenating encoder features directly into the decoder [37].

3.5.3.1 Skip Connections

To preserve fine-grained details during the reconstruction process, skip connections are employed between corresponding layers of the encoder and decoder. These connections combine high-resolution spatial information from the encoder with the semantic features in the decoder, improving the model's ability to segment fine details, such as narrow vineyard rows.

$$Z_l = Concat(X_l^{encoder}, X_l^{decoder})$$

Where $X_l^{encoder}$ and $X_l^{decoder}$ are the feature maps from the encoder and decoder at the same spatial resolution.
This is illustrated in Figure 11.

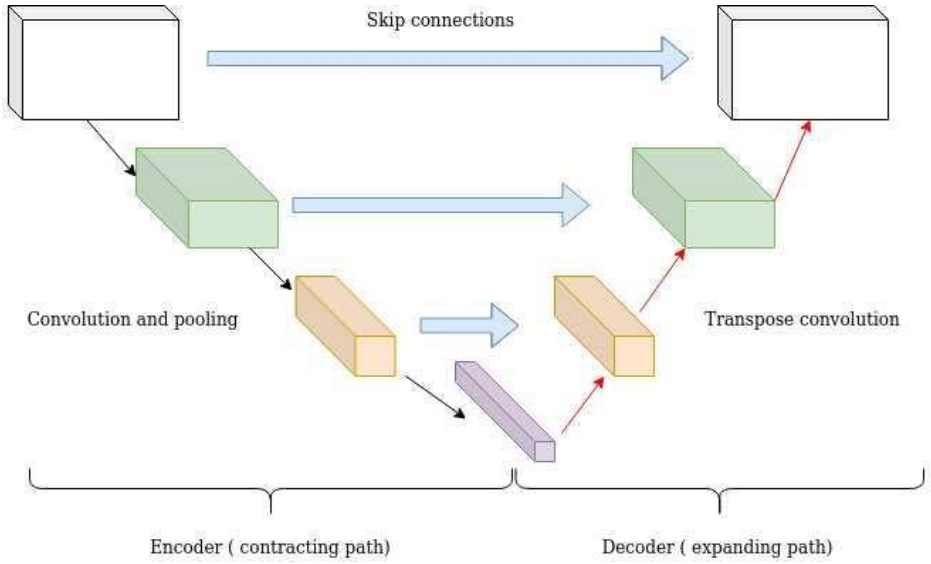


Figure 11: Skip connections in a symmetric encoder-decoder (U-Net-style) architecture [36].

3.5.3.2 Atrous (Dilated) Convolutions

Atrous convolutions are used in advanced models like DeepLab [22] to increase the receptive field of the network without reducing spatial resolution. This allows the model to capture multi-scale contextual information, which is particularly important in segmenting objects of varying sizes in UAV imagery.

A dilated convolution modifies the standard convolution by introducing a dilation rate d, which defines the spacing between kernel elements:

$$Y(i,j) = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} w(m,n) * X(i + dm,\ j + dn) + b$$

Where:

- $Y(i,j)$ is the output feature at the position (i,j).
- X(i+dm, j+dn) is the input value at position (i+dm, j+dn).
- $w(m,n)$ is the weight of the kernel at position (m,n).
- b is the bias term.
- k defines the size of the kernel (e.g., 3×3).
- d is the dilation rate.

Atrous convolutions are particularly useful for segmenting objects at multiple scales, as they can capture contextual information from a wider area.

3.5.3.3 Attention Mechanisms

The Transformer is a neural network architecture originally developed for processing sequential data such as text, used in tasks like translation and summarization. Unlike earlier sequential models such as recurrent neural networks (RNNs), which process a sequence one element at a time, a Transformer processes the entire sequence at once - both speeding up computation and making it easier to capture relationships between elements that are far apart in the sequence. At its core is the self-attention mechanism, which allows each element of the sequence to weigh the relevance of every other element when computing its own representation. Since this mechanism has no inherent notion of order, positional encodings are added to the input to preserve sequence order. The original architecture consists of an encoder, which processes the input, and a decoder, which generates the output, although many later models use only one of the two components - for example, BERT uses only the encoder, while GPT uses only the decoder. Although Transformers were originally developed for language tasks, the same self-attention mechanism has since been adapted to computer vision by treating fixed-size image patches as the input tokens, forming the basis of the attention-based segmentation models discussed below.

Attention mechanisms, including self-attention, have recently been integrated into segmentation models to focus on relevant regions of the image while ignoring irrelevant parts [22]. These mechanisms improve the model's ability to differentiate between vineyard rows and the surrounding environment.
Self-attention computes the importance of each pixel relative to others:

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

Where:

- Q, K, V are the query, key, and value matrices derived from the input features.
- d_k is the dimensionality of the key matrix.
- The softmax function normalizes the attention weights.

This mechanism allows the model to capture long-range dependencies and focus on relevant vineyard rows, even in complex backgrounds.

In this thesis, we focus on two architectures, U-Net and Dense U-Net, each offering a different strategy for preserving spatial detail while extracting increasingly abstract features.

3.5.4 U-Net

Primarily developed for medical imaging, U-Net's architecture features a "U"-shaped design, as illustrated in Figure 12. It consists of a symmetric encoder-decoder structure that includes shortcuts, i.e., skip connections, to capture detailed contextual information and retain critical data throughout the image processing phases. This architecture is ideal for the precise localization needed to navigate UGVs in vineyard rows, ensuring that no detail is missed in vine density and row alignment. The network takes as input an image (in our application, a drone capture of the vine) and as a ground truth, it takes the same dimensional mask of the feature of interest (vine rows) that should be reconstructed as an output.

The U-Net architecture starts with a contracting path, which is the encoder part. This path is responsible for capturing the context of the input image through a series of convolutional and pooling layers. Each layer in the contracting path reduces the spatial dimensions of the input while increasing the depth, allowing the network to extract high-level features. This process helps in understanding the broader context of the image.

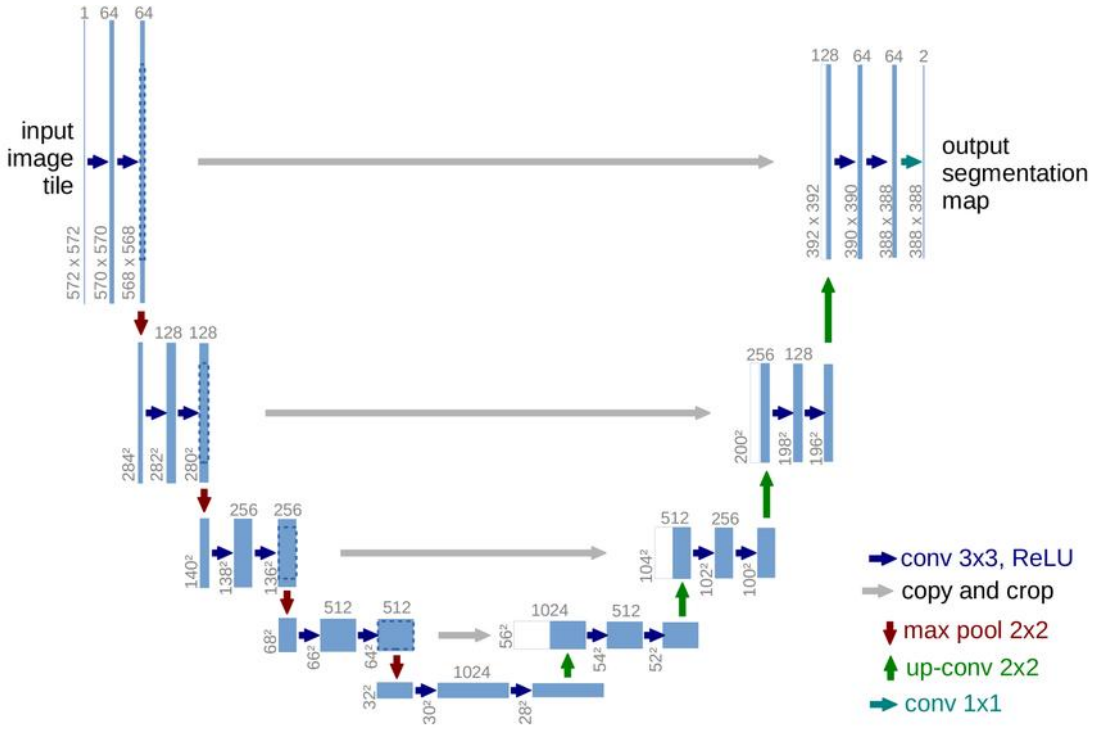


Figure 12: U-Net Architecture *(Ronneberger et al., 2015)*, with encoder half on the left and a decoder on the right, and skip connections indicated with the arrows between the two.

U-Net employs an expansive path, or decoder, following the encoder, which symmetrically up-samples the features back to the original image size. This path consists of up-convolutions and concatenations with corresponding feature maps from the contracting path. These concatenations, or skip connections, are pivotal as they merge the high-resolution features from the contracting path with the upsampled output, ensuring that fine details are preserved. This approach mitigates the loss of spatial information that typically occurs in deep networks, making U-Net exceptionally effective for tasks requiring precise localization and boundary delineation.

3.5.5 Dense U-Net

Dense U-Net follows the same overall encoder-decoder structure as U-Net, but replaces each convolutional block with a densely connected block, in which every layer receives the concatenated outputs of all preceding layers within the same block rather than only the output of the immediately preceding one. This encourages feature reuse throughout the network: low-level details extracted early in a block remain directly accessible to later layers instead of having to pass through every intermediate transformation. For a task such as vine row segmentation, where the distinguishing signal between a row and the surrounding soil can be subtle, this denser flow of information helps the network retain fine-grained detail that might otherwise be diluted as it propagates through successive layers. As in U-Net, skip

connections between the encoder and decoder are retained to preserve spatial resolution during upsampling. The overall architecture is illustrated in Figure 13.

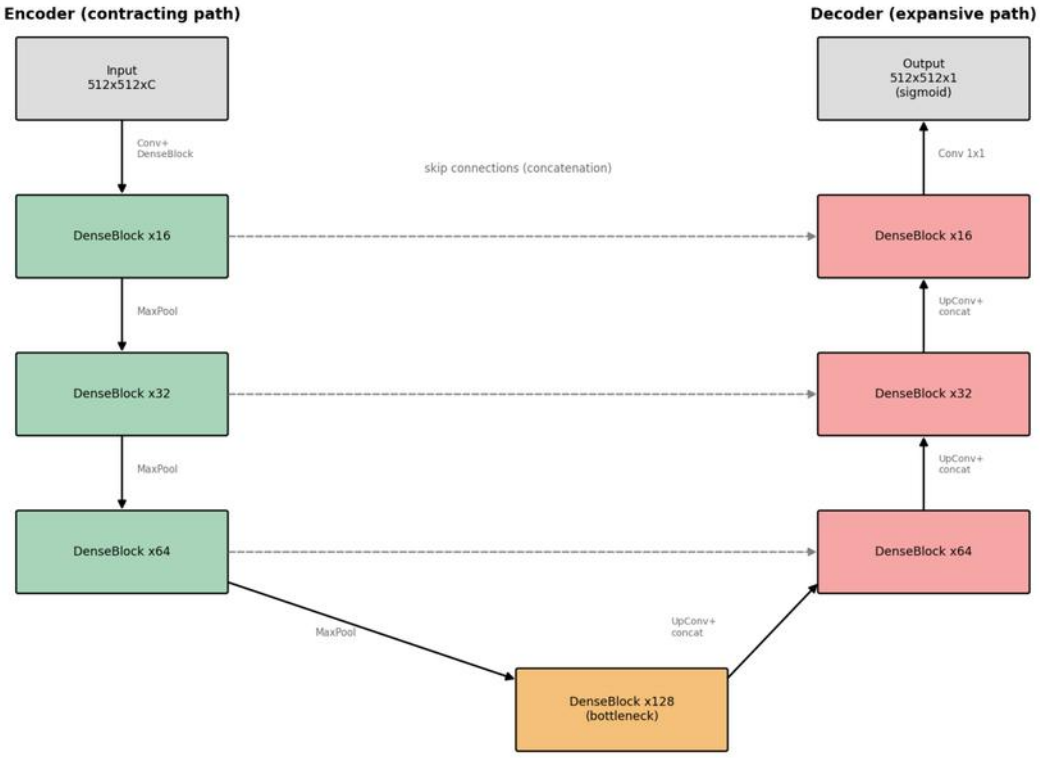


Figure 13: Dense U-Net architecture, with the encoder (contracting path) on the left and the decoder (expansive path) on the right. Each block represents a densely connected block operating at a given spatial resolution, with resolution reduced via max pooling in the encoder and restored via up-convolution in the decoder. Skip connections (dashed arrows) link encoder and decoder blocks of matching resolution, and the bottleneck block at the bottom connects the two paths.

3.5.6 Dropout

Dropout is a regularization technique in which a randomly selected fraction of neuron activations is set to zero at each training step. This prevents the network from relying too heavily on any single neuron or feature, forcing it to learn more redundant, distributed representations. It directly addresses overfitting - the phenomenon in which a model fits the training data too closely, including its noise, at the expense of generalizing to unseen data, typically manifesting as strong training performance alongside markedly worse validation performance [32].

3.5.7 Optimization

Training a neural network consists of iteratively adjusting its weights to minimize a loss function, typically using gradient descent: at each step, the gradient

of the loss with respect to the weights is computed, and the weights are updated in the opposite direction. Adam (Adaptive Moment Estimation) [33] is a widely used extension of gradient descent that maintains a separate, adaptive learning rate for each weight, based on running estimates of the first moment (mean) and second moment (uncentered variance) of its past gradients, combining the benefits of momentum-based acceleration with per-parameter adaptive scaling.

3.5.8 Batch Normalization

Batch normalization normalizes the activations of a layer - rescaling them to zero mean and unit variance - using statistics computed over each mini-batch during training. This stabilizes and speeds up training by reducing the sensitivity of later layers to shifts in the distribution of earlier layers' outputs, and is applied after nearly every convolutional layer in both U-Net and Dense U-Net in this work [34].

3.6 Evaluation Metrics

Semantic segmentation metrics are essential for quantifying the performance of models in pixel-level classification tasks. These metrics provide insights into the model's ability to segment different regions in an image accurately and are critical for comparing various model architectures and configurations. Below, the most commonly used metrics are described in detail.

3.6.1 Intersection over Union (IoU)

The Intersection over Union (IoU), also known as the Jaccard Index, is a measure of the overlap between the predicted segmentation mask and the ground truth mask. It evaluates how well the predicted region aligns with the ground truth by calculating the intersection ratio to the union of the two sets of pixels.

IoU values range from 0 to 1, where 1 indicates perfect overlap between the predicted and ground truth masks. IoU is computed for each class separately, and the mean IoU (mIoU) is taken as the average across all classes.

Mathematical Formula:

$$IoU(P, G) = \frac{|P \cap G|}{|P \cup G|}$$

Where:

- P is the predicted mask (pixels classified as belonging to the target class by the model).
- G is the ground truth mask (pixels labeled as belonging to the target class in the dataset).

- $|P \cap G|$ is the number of pixels correctly predicted as belonging to the target class (true positives).
- $|P \cup G|$ is the total number of pixels in either P or G (union of true positives, false positives, and false negatives).

3.6.2 Dice Coefficient

The Dice Coefficient, also known as the Sørensen-Dice Index, measures the similarity between the predicted and ground truth masks. It emphasizes the overlap between the two masks and penalizes discrepancies, such as false positives and false negatives.

The Dice Coefficient ranges from 0 to 1, where 1 indicates perfect agreement between the predicted and ground truth masks. Higher Dice scores imply better segmentation performance.

Mathematical Formula:

$$Dice = \frac{2|P \cap G|}{|P| + |G|}$$

Where:

- P is the predicted mask.
- G is the ground truth mask.
- $|P|$ the number of pixels in the predicted mask (total predicted positives).
- $|G|$ is the number of pixels in the ground truth mask (total actual positives).
- $|P \cap G|$ is the number of pixels correctly classified as belonging to the target class.

3.6.3 Pixel Accuracy

Pixel Accuracy measures the proportion of correctly classified pixels (both positive and negative) to the total number of pixels in the image. It provides a global measure of model performance over the entire image.

Pixel Accuracy values range from 0 to 1, with 1 indicating all pixels are correctly classified.

$$Pixel\ Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Where:

- TP is True positives (pixels correctly predicted as the target class).

- TN is True negatives (pixels correctly predicted as background or other classes).
- FP is False positives (background pixels incorrectly classified as the target class).
- FN is False negatives (target class pixels incorrectly classified as background).

3.6.4 Precision

Precision measures the proportion of pixels predicted as belonging to the vine-row class that are actually correct, quantifying how many of the predicted row pixels correspond to real rows rather than false detections.

Precision values range from 0 to 1, with 1 indicating that every pixel predicted as a row is indeed part of a row.

$$Precision = \frac{TP}{(TP + FP)}$$

Where TP and FP are as defined in Section 3.6.3.

3.6.5 Recall

Recall, also referred to as Sensitivity, measures the proportion of actual row pixels that were correctly identified by the model, quantifying how much of the true row area was successfully detected.

Recall values range from 0 to 1, with 1 indicating that every true row pixel was correctly detected.

$$Recall = \frac{TP}{TP+FN}$$

Where TP and FN are as defined in Section 3.6.3.

3.6.6 F1-Score

The F1-score is the harmonic mean of Precision and Recall, providing a single measure that balances the cost of false positives against the cost of false negatives.

F1-score values range from 0 to 1, with 1 indicating perfect precision and recall simultaneously.

$$F1 = 2 * \frac{(Precision \times Recall)}{(Precision + Recall)}$$

Where Precision and Recall are as defined in Sections 3.6.4 and 3.6.5.

3.6.7 Specificity

Specificity measures the proportion of actual background pixels that were correctly identified as background, quantifying the model's ability to avoid false positive row predictions.

Specificity values range from 0 to 1, with 1 indicating that every true background pixel was correctly classified.

$$Specificity = \frac{TN}{(TN + FP)}$$

Where TN and FP are as defined in Section 3.6.3.

The motivation of each metric, as well as its strengths and weaknesses, are detailed in Table 1.

Metric	Purpose	Strengths	Weaknesses
IoU	Quantifies the overlap between predicted and ground truth masks.	Balances false positives and false negatives.	Sensitive to small regions or imbalances.
Dice Coefficient	Emphasizes overlap and penalizes false positives/negatives.	Effective for imbalanced classes.	Less interpretable for minor class errors.
Pixel Accuracy	Measures overall classification performance across all pixels.	Simple and intuitive metric.	Fails to highlight issues in imbalanced data.
Precision	Measures how many predicted row pixels are actually correct.	Directly penalizes false positives.	Ignores missed detections (false negatives).
Recall	Measures how much of the true row area was detected.	Directly penalizes missed detections.	Ignores false positives; can be inflated by over-predicting.

F1-Score	Balances precision and recall into a single score.	Useful single summary when both false positives and negatives matter.	Can mask which of precision or recall is driving a low score.
Specificity	Measures how well background pixels are correctly identified.	Highlights false-positive avoidance on the majority class.	Can appear high by default in imbalanced data even with poor row detection.

Table 1: Comparison between different evaluation metrics

4. Experiments and Results

4.1 Experimental Setup

Each of the four spectral channel combinations described in Section 3.2 was used to train and evaluate both segmentation architectures independently, allowing their performance to be compared under identical training conditions while varying only the spectral input.

For each channel combination, the dataset was split into training, validation, and test sets in a 70/20/10 ratio, yielding 11,544 training, 3,296 validation, and 1,656 test image - label pairs (16,496 patches in total, pooled across the three simulated altitude scales and eight augmentation variants described in Section 3.4). Since every original patch was augmented through rotation and flipping prior to splitting, augmented variants of the same patch were kept together within a single split, ensuring that no transformed copy of a training image could appear in the validation or test sets.

Both architectures were trained using the Adam optimizer with an initial learning rate of 0.001, reduced by half whenever validation loss failed to improve for two consecutive epochs, down to a minimum of 1e-6. All model weights were randomly initialized (He normal for U-Net, Glorot uniform for Dense U-Net) rather than pretrained, so each model learned entirely from the training data used in this study. Training was limited to a maximum of 10 epochs, with early stopping applied if validation loss did not improve for six consecutive epochs, restoring the weights from the best-performing epoch. A dropout rate of 0.1 was applied throughout both networks to reduce overfitting (Section 3.5.6).

As the vine-row class occupies a minority of pixels in most patches, a combined binary cross-entropy and Dice loss was used for training rather than binary cross-entropy alone. Specifically, across the pooled test set, row pixels account for

approximately 22% of all pixels, with roughly one in five test patches containing no row pixels at all. Under this imbalance, metrics computed primarily on the majority class - such as pixel accuracy and specificity - can appear misleadingly high even for a classifier that performs poorly at detecting rows. Dice coefficient, IoU, precision, recall, and F1-score are therefore emphasized as the primary measures of segmentation quality in this work, with pixel accuracy and specificity reported for completeness (see Table 1 for a discussion of each metric's sensitivity to class imbalance).

Due to GPU memory constraints, batch size of 2 was used across experiments. Training was performed on a SLURM-managed GPU computing cluster, using a single NVIDIA GeForce RTX 2080 Ti GPU (11 GB GDDR6) per job, with 4 CPU cores (Intel Xeon Gold 6226) and 10 GB of RAM allocated, running Ubuntu Server 20.04 LTS with CUDA 11.2 and mixed-precision computation - in which most operations use lower-precision (16-bit) floating-point numbers while numerically sensitive operations retain full (32-bit) precision, reducing memory usage and training time. All eight channel-and-architecture configurations were trained sequentially within this allocation.

Following training, each model was evaluated on its corresponding test set using Intersection over Union, Dice coefficient, pixel accuracy, precision, recall, F1-score, and specificity, computed per patch and aggregated across the test set.

All models were implemented and trained using TensorFlow with its Keras API. Experiment tracking, including the logging of evaluation metrics and trained models for each configuration, was handled using MLflow. Training was carried out on a SLURM-managed GPU computing cluster, allowing multiple channel-and-architecture configurations to be trained as separate jobs.

4.2 Results and Analysis

Table 2 reports the test-set performance of both architectures across all four spectral channel combinations introduced in Section 3.3. As expected for binary segmentation, the Dice coefficient and F1-score are computed from an algebraically equivalent formulation and therefore coincide numerically throughout the table, both are reported for consistency with Table 1.

Channel combination	Architecture	IoU	Dice	Pixel Acc.	Precision	Recall	F1	Specificity
RGB-NIR	U-Net	0.7995	0.8886	0.9520	0.9172	0.8616	0.8886	0.9778
RGB-NIR	Dense U-Net	0.7984	0.8879	0.9499	0.8815	0.8944	0.8879	0.9657
RGB-RE	U-Net	0.7983	0.8878	0.9519	0.9195	0.8583	0.8878	0.9786
RGB-RE	Dense U-Net	0.7981	0.8877	0.9521	0.9252	0.8532	0.8877	0.9803
Green-RE-NIR	U-Net	**0.8052**	**0.8921**	**0.9530**	0.9067	0.8780	**0.8921**	0.9743
Green-RE-NIR	Dense U-Net	0.7831	0.8783	0.9472	0.8972	0.8602	0.8783	0.9720
RGB	U-Net	0.7519	0.8584	0.9423	0.9392	0.7904	0.8584	0.9855
RGB	Dense U-Net	0.5916	0.7434	0.9073	**0.9589**	0.6070	0.7434	**0.9926**

Table 2: Test-set evaluation metrics for all eight trained configurations.

Figure 14 visualizes the Dice coefficient and IoU across all eight configurations, grouped by spectral channel combination.

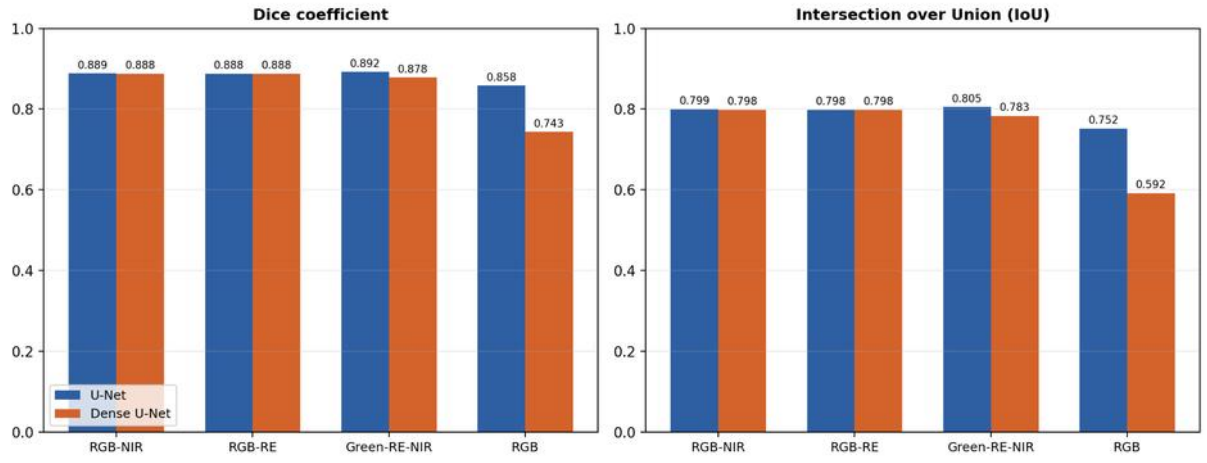


Figure 14: Test-set Dice coefficient and IoU across all eight trained configurations, grouped by spectral channel combination and colored by architecture.

Effect of spectral channel combination

The three combinations that include a near-infrared or red-edge band, marked as RGB-NIR, RGB-RE, and Green-RE-NIR - achieved consistently high and closely

clustered performance (Dice between 0.878 and 0.892, IoU between 0.783 and 0.805) regardless of architecture. This is consistent with the spectral properties discussed in Section 3.3: healthy vine canopy reflects strongly in the near-infrared and red-edge regions relative to soil and inter-row background, which produces a stronger and more separable signal for the segmentation model to exploit than the visible spectrum alone provides. The pure RGB combination, which lacks any such band, achieved a visibly lower Dice coefficient with both architectures (0.858 with U-Net, 0.743 with Dense U-Net), supporting the interpretation that access to a near-infrared or red-edge band is the primary driver of segmentation quality in this task, more so than the specific choice of architecture.

Effect of architecture

U-Net outperformed Dense U-Net on every one of the four channel combinations tested, though the size of the gap varied considerably: negligible for RGB-NIR and RGB-RE (differences below 0.001 in Dice), moderate for Green-RE-NIR (0.014), and substantial for RGB (0.115). This pattern suggests that the additional representational capacity of the densely connected blocks did not translate into a modelling advantage for this task, and that Dense U-Net's larger memory footprint and deeper block structure may have made it more susceptible to the training instability discussed below.

The RGB + Dense U-Net result

This configuration stands out as a clear outlier, with the lowest scores on every metric except precision and specificity, where it is in fact the highest of all eight configurations (precision 0.9589, specificity 0.9926). Combined with its markedly lower recall (0.607, the lowest of all configurations), this indicates that the model did not become confused about the appearance of a vine row; when it predicted the row class, it was correct more often than any other configuration. Rather, it became overly conservative, failing to detect a substantial share of the true row area. This asymmetric failure mode - high precision, low recall - is more consistent with a model whose training was disrupted before it could converge to a confident decision boundary than with a model that learned an incorrect representation of the input.

Training dynamics and overfitting

Figure 15 compares the per-epoch training and validation curves (loss and Dice coefficient) of the best-performing configuration (Green-RE-NIR + U-Net) against the RGB + Dense U-Net run. In the well-behaved run, training and validation curves remain close throughout all ten epochs, with the validation curve showing the ordinary epoch-to-epoch noise expected from a limited validation set but no systematic widening gap - indicating that overfitting was not a significant concern under the regularization settings used (dropout of 0.1 and early stopping, Section 4.1). The RGB + Dense U-Net run shows the same healthy behaviour through epoch eight, with training and validation curves tracking closely. In the final two epochs,

however, validation loss rises sharply (from 0.325 to 0.569) and validation Dice collapses (from 0.839 to 0.694), while the training curve continues its smooth, unremarkable trend. This late, abrupt divergence - rather than a gradual widening gap from early in training - is not the signature typically associated with overfitting, and is instead consistent with an external disruption to the training process late in the run rather than a genuine failure of the model to generalize. This is discussed further as a limitation below.

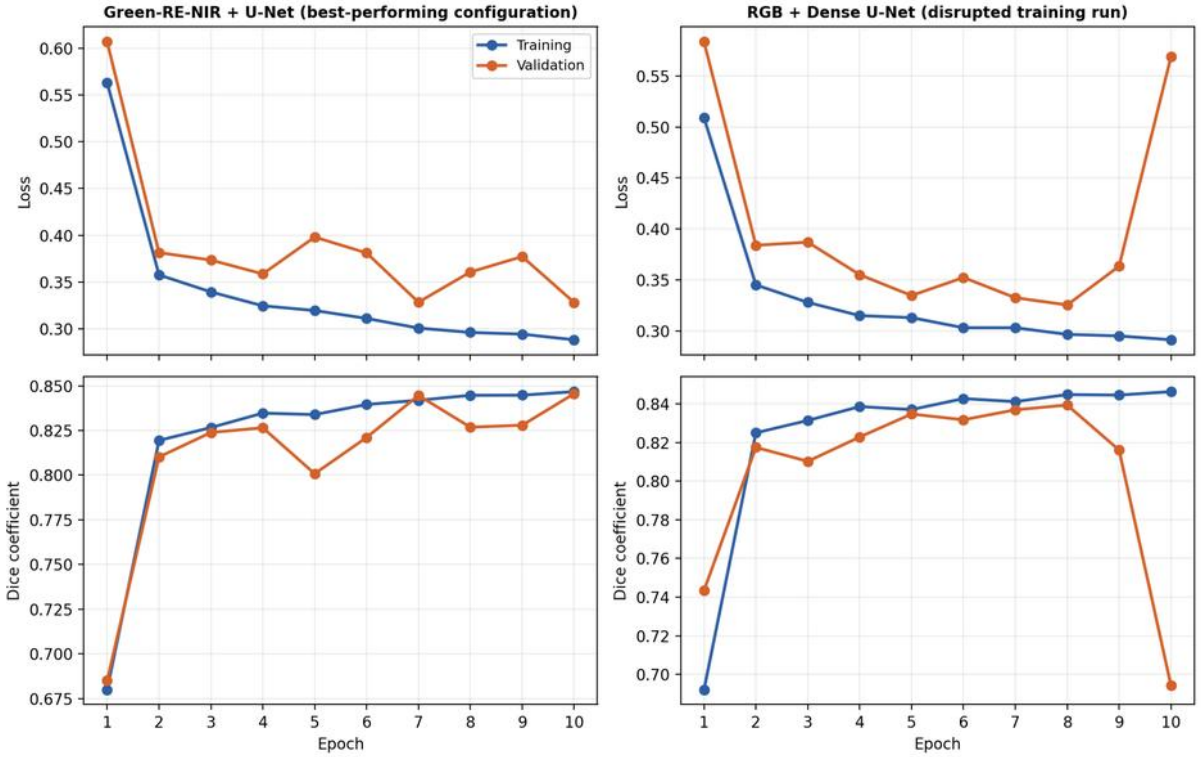


Figure 15: Per-epoch training and validation curves (loss and Dice coefficient) for the best-performing configuration (Green-RE-NIR + U-Net, left) compared against the RGB + Dense U-Net run (right), illustrating the late-epoch divergence in the latter.

Qualitative results

Figure 16 shows predictions from the best-performing configuration (Green-RE-NIR + U-Net) on five test patches spanning a range of ground-truth row coverage, selected by percentile rank rather than manual curation. For patches containing no rows, isolated row segments, or a single large row spanning most of the patch, the predicted mask closely matches the ground truth in both presence and approximate geometry. The intermediate examples illustrate the two main sources of residual error observed across the test set: for patches containing several oblique row segments, the model correctly locates each region but does not always reproduce its precise boundary or extent; and for a patch containing a single small, irregularly shaped row occupying a minor part of the patch, the model detects only a fragment of the true row area. This latter behaviour is consistent with the per-image variability discussed below, and indicates that segmentation quality degrades on small or atypically shaped row instances even in the best-performing configuration.

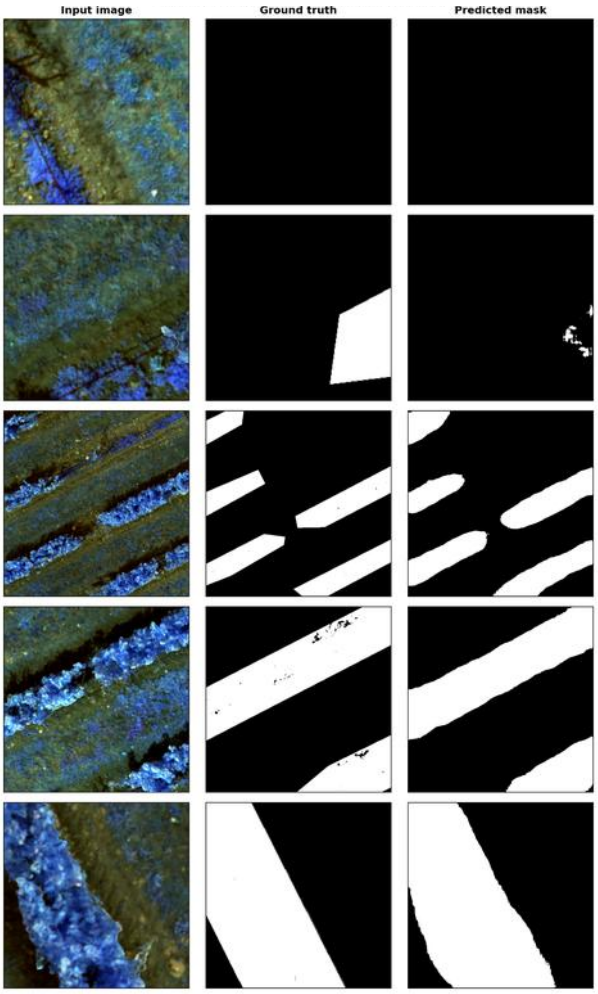


Figure 16: Qualitative segmentation examples from the best-performing configuration (Green-RE-NIR + U-Net) on five test patches spanning a range of ground-truth row coverage. Columns show, left to right: input image, ground truth mask, and predicted mask.

Per-image variability

In addition to the pooled test-set metrics in Table 2, per-patch metrics were also computed and averaged across the 1,656 test images for each configuration. This per-image analysis confirms the overall ranking of configurations but reveals substantially larger variability for RGB + Dense U-Net (mean per-image IoU of 0.583 with a standard deviation of 0.308) compared to the other seven configurations (standard deviations of 0.26-0.29 on a higher mean), indicating that its lower aggregate performance stems partly from a subset of images segmented considerably worse than average, rather than a uniformly moderate degradation across the entire test set.

To assess whether segmentation quality is sensitive to the drone altitude simulated through the patch-scale augmentation described in Section 3.4, the best-performing model's test-set predictions were also broken down by source patch size. Table 3 reports pooled evaluation metrics separately for patches upsampled from 256×256 (simulating lower altitude), native 512×512 patches, and patches downsampled from 1024×1024 (simulating higher altitude). For computational feasibility, the native and 256×256 groups were evaluated on a random sample of 300 test patches each, while

the 1024×1024 group - comprising only 72 test patches in total - was evaluated in full.

Simulated altitude	Source scale	n	IoU	Dice	Precision	Recall	Specificity
Lower altitude	256→512 (upsampled)	300	0.805	0.892	**0.911**	0.874	0.974
Native	512 (unmodified)	300	**0.820**	**0.901**	0.905	0.898	**0.975**
Higher altitude	1024→512 (downsampled)	72	0.777	0.875	0.844	**0.908**	0.970

Table 3: Test-set metrics for the best-performing configuration (Green-RE-NIR + U-Net), broken down by simulated flight altitude.

Native-resolution patches achieved the highest scores, with the upsampled (lower-altitude) patches close behind and the downsampled (higher-altitude) patches showing the largest, though still modest, drop in IoU and Dice. This asymmetry is consistent with the nature of the two resizing operations: upsampling from 256×256 interpolates from genuinely finer real-world detail, whereas downsampling from 1024×1024 averages multiple real pixels into one, discarding fine boundary information between rows and inter-row soil that the model relies on. Figure 17 illustrates this qualitatively for one representative test patch per altitude group, showing the predicted mask decomposed into true positive, false positive, and false negative regions: the proportion of boundary-region errors (false positives and false negatives) visibly increases at the higher simulated altitude, while the core of each detected row remains correctly classified across all three altitude conditions. This indicates that the model's overall ability to locate vine rows is robust across the tested range of simulated altitudes, with boundary precision being the primary factor affected by reduced source resolution.

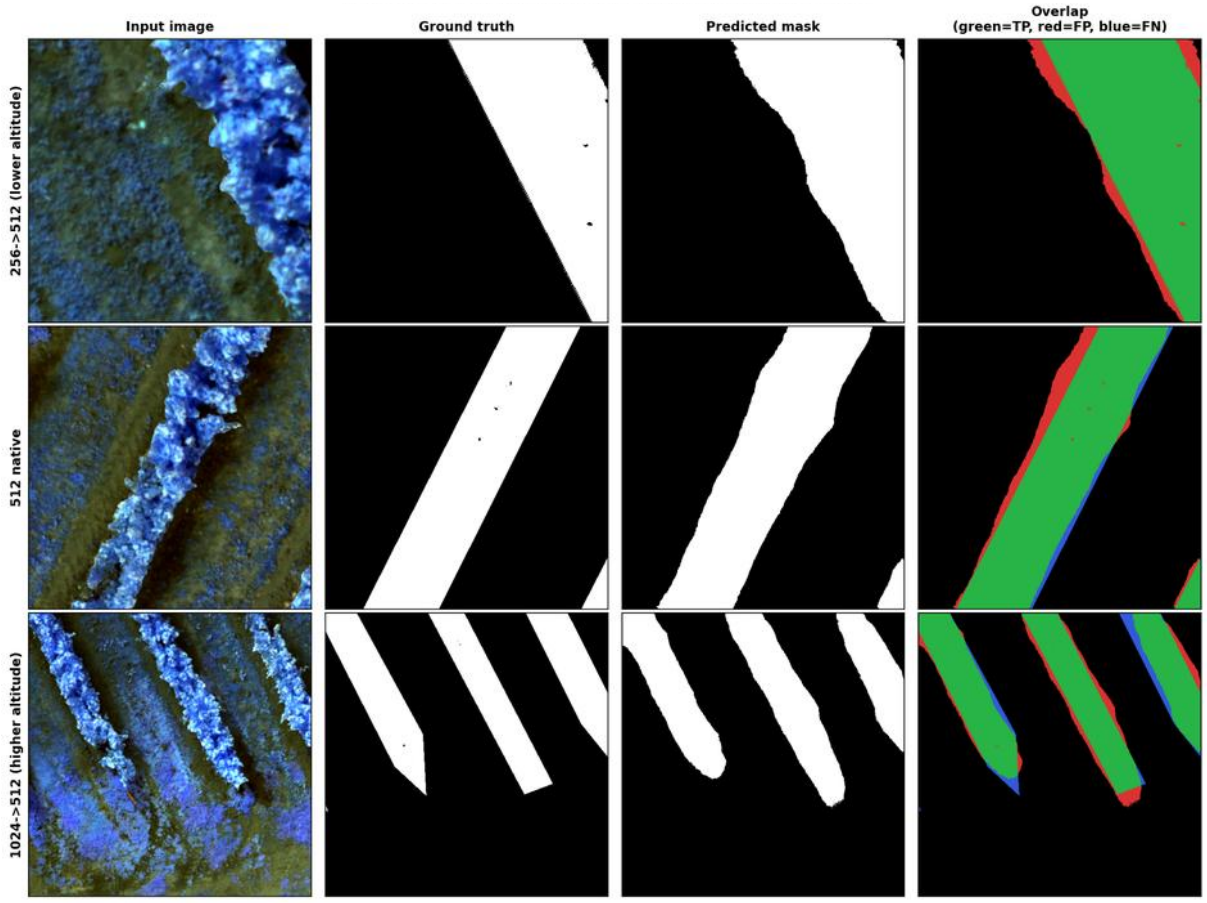


Figure 17: Prediction overlap by simulated flight altitude, for one representative test patch per altitude group (Green-RE-NIR + U-Net). Rows correspond to lower altitude (256→512, top), native resolution (middle), and higher altitude (1024→512, bottom). Columns show, left to right: input image, ground truth, predicted mask, and the prediction-ground truth overlap (green = true positive, red = false positive, blue = false negative).

5. Conclusion and Future Work

Experimental results presented in Chapter 4 show that spectral bands sensitive to vegetation health, near-infrared and red-edge, enable substantially more accurate row segmentation than visible-spectrum imagery alone, with the best configuration reaching a Dice coefficient of 0.89 on held-out test data. These findings offer a concrete basis for selecting sensor and model configurations when deploying such a system for real-world UGV navigation.

Precision agriculture increasingly depends on the ability of autonomous systems to perceive their environment accurately enough to act on it. This thesis addressed one concrete instance of that challenge: teaching a segmentation model to reliably distinguish vineyard rows from surrounding soil and vegetation in UAV imagery, as a foundation for guiding an unmanned ground vehicle through the vineyard's inter-row spaces. Two encoder-decoder architectures, U-Net and Dense U-Net, were trained and evaluated across four spectral channel combinations: RGB,

RGB-NIR, RGB-RE, and Green-RE-NIR - using manually digitized vine-row polygons as ground truth.

The results point to a clear, actionable conclusion: the presence of a near-infrared or red-edge band, not the choice between U-Net and Dense U-Net, is what determines segmentation quality in this task. All three channel combinations incorporating such a band reached Dice coefficients above 0.87, tightly clustered regardless of architecture, while the purely visible-spectrum RGB combination performed measurably worse - consistent with the well-established spectral distinctiveness of healthy vine canopy in the near-infrared and red-edge regions relative to soil and inter-row background. Between the two architectures, U-Net matched or outperformed the more complex Dense U-Net on every channel combination tested, suggesting that for this task, architectural complexity is a less productive investment than sensor selection.

These findings are not without qualification. Each configuration was trained once rather than across multiple random seeds, so the narrow spread among the three near-infrared/red-edge combinations (Dice coefficients between 0.878 and 0.892) should be read as a close grouping rather than a definitive ranking. Training was also carried out on a shared GPU cluster, and the RGB + Dense U-Net run in particular was disrupted late in training - visible as a sharp divergence between training and validation performance in its final two epochs - meaning its comparatively low score likely understates what that combination could achieve under stable conditions. Evaluation was further limited to a single study site and acquisition campaign, leaving generalization to other vineyards, growth stages, and seasons untested.

Addressing these limitations - repeated training runs for statistical confidence, a clean re-run of the disrupted configuration, and evaluation across additional sites - would sharpen the conclusions drawn here. But the core result already stands on solid ground: reliable, high-accuracy row detection is achievable from UAV imagery with a comparatively simple architecture, provided the sensor captures the right part of the spectrum. Coupled with a path-planning layer and validated through field trials, this positions row segmentation as a practical, near-term building block for autonomous navigation in precision viticulture - bringing the vineyard a step closer to the technology-assisted future imagined at the outset of this work, without losing sight of the tradition it is built on.

Bibliography

[1] Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. arXiv preprint arXiv:1505.04597.

[2] Wine in Art: Ancient History. Wine & Other Stories. January 29, 2023.

[3] Case, J. H. (2019). New Ways Drones Are Changing Vineyard Management. SevenFifty Daily.

[4] Ennouri, K., Smaoui, S., Gharbi, Y., Cheffi, M., Ben Braiek, O., Ennouri, M., Triki, M.A. (2021). Usage of Artificial Intelligence and Remote Sensing as Efficient Devices to Increase Agricultural System Yields. Journal of Food Quality, 2021, 6242288. https://doi.org/10.1155/2021/6242288

[5] LeCun, Y., Bengio, Y., Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. https://doi.org/10.1038/nature14539

[6] Minaee, S., Boykov, Y., Porikli, F., Plaza, A., Kehtarnavaz, N., Terzopoulos, D. (2022). Image segmentation using deep learning: A survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(7), 3523–3542. https://doi.org/10.1109/TPAMI.2021.3059968

[7] Bello, S.A., Yu, S., Wang, C., Adam, J.M., Li, J. (2020). Review: Deep Learning on 3D Point Clouds. Remote Sensing, 12(11), 1729. https://doi.org/10.3390/rs12111729

[8] Gautron, R., Maillard, O., Preux, P., Corbeels, M., Sabbadin, R. (2022). Reinforcement learning for crop management support: Review, prospects and challenges. Computers and Electronics in Agriculture, 200, 107182. https://doi.org/10.1016/j.compag.2022.107182

[9] Huang, Y., Thomson, S., Lan, Y., Maas, S. (2010). Multispectral imaging systems for airborne remote sensing to support agricultural production management. International Journal of Applied Business and Engineering (IJABE), Article ID 181.

[10] Gallo, R., Ristorto, G., Daglio, G., Massa, N., Berta, G., Lazzari, M., Mazzetto, F. (2017). New Solutions for the Automatic Early Detection of Diseases in Vineyards through Ground Sensing Approaches Integrating LiDAR and Optical Sensors. Chemical Engineering Transactions, 58.

[11] Nguyen, T., Hoang, T., Pham, M., Vu, T., Nguyen, T., Huynh, Q., Jo, J. (2020). Monitoring agriculture areas with satellite images and deep learning. Applied Soft Computing, 95, 106565. https://doi.org/10.1016/j.asoc.2020.106565

[12] Boubrahimi, S.F., Aydin, B., Kempton, D., Angryk, R. (2016). The sun in different wavelengths: SOLEV: A Video Generation Framework for Solar Events from Mixed Data Sources (Demo Paper). https://doi.org/10.1145/2996913.2996963

[13] Chang, Z., Li, H., Chen, D., Liu, Y., Zou, C., Chen, J., Han, W., Liu, S., Zhang, N. (2023). Crop Type Identification Using High-Resolution Remote Sensing Images Based on an

Improved DeepLabV3+ Network. Remote Sensing, 15(21), 5088. https://doi.org/10.3390/rs15215088

[14] Yu, S., Liu, X., Tan, Q., Wang, Z., Zhang, Z. (2024). Sensors, systems, and algorithms of 3D reconstruction for smart agriculture and precision farming: A review. Computers and Electronics in Agriculture. https://doi.org/10.1016/j.compag.2024.109229

[15] Miller, M. (2017). Point cloud smart mapping in 3D with Scene Viewer. ArcGIS Blog.

[16] Spectral Signature Cheatsheet in Remote Sensing. GISGeography. https://gisgeography.com/spectral-signature/

[17] OpenCV documentation. https://docs.opencv.org/4.x/index.html

[18] Keys, R. (1981). Cubic convolution interpolation for digital image processing. IEEE Transactions on Acoustics, Speech, and Signal Processing, 29(6), 1153–1160. https://doi.org/10.1109/TASSP.1981.1163711

[19] Dong, W. (2018). What is OpenCV's INTER_AREA Actually Doing? Medium. https://medium.com/@wenrudong/what-is-opencvs-inter-area-actually-doing-282a626a09b3

[20] Guo, Y., Liu, Y., Georgiou, T., et al. (2018). A review of semantic segmentation using deep neural networks. International Journal of Multimedia Information Retrieval, 7, 87–93. https://doi.org/10.1007/s13735-017-0141-z

[21] Chen, L., Papandreou, G., Murphy, K., Yuille, A. (2015). Semantic Image Segmentation with Deep Convolutional Nets and Fully Connected CRFs. arXiv:1412.7062.

[22] Zhang, J., Chen, X., Yang, B., Guan, Q., Chen, Q., Chen, J., Wu, Q., Xie, Y., Xia, Y. (2025). Advances in attention mechanisms for medical image segmentation. Computer Science Review, 55, 100721. https://doi.org/10.1016/j.cosrev.2024.100721

[23] Elharrouss, O., Akbari, Y., Almaadeed, N., Al-Maadeed, S. (2022). Backbones-Review: Feature Extraction Networks for Deep Learning and Deep Reinforcement Learning Approaches. arXiv:2206.08016.

[24] Biglia, A., Zaman, S., Gay, P., Ricauda Aimonino, D., Comba, L. (2022). 3D point cloud density-based segmentation for vine rows detection and localisation. Computers and Electronics in Agriculture, 199, 107166. https://doi.org/10.1016/j.compag.2022.107166

[25] Hatfield, J.L., Gitelson, A.A., Schepers, J.S., Walthall, C.L. (2008). Application of Spectral Remote Sensing for Agronomic Decisions. Agronomy Journal, 100, S-117–S-131.

[26] Lu, Y., Chen, D., Olaniyi, E., Huang, Y. (2022). Generative adversarial networks (GANs) for image augmentation in agriculture: A systematic review. Computers and Electronics in Agriculture, 200, 107208. https://doi.org/10.1016/j.compag.2022.107208

[27] Comba, L., Gay, P., Primicerio, J., Ricauda Aimonino, D. (2015). Vineyard Detection from Unmanned Aerial Systems Images. Computers and Electronics in Agriculture, 114, 78–87. https://doi.org/10.1016/j.compag.2015.03.011

[28] Vera-Esmeraldas, A., Pizarro-Otáza, S., Labbé, M., Rojo, F., Salazar, F. (2025). UAV-Based Spectral and Thermal Indices in Precision Viticulture: A Review of NDVI, NDRE, SAVI, GNDVI, and CWSI. Agronomy, 15(11), 2569. https://doi.org/10.3390/agronomy15112569

[29] Rouse, J.W., Haas, R.H., Schell, J.A., Deering, D.W. (1974). Monitoring Vegetation Systems in the Great Plains with ERTS. Third ERTS-1 Symposium, NASA SP-351, Washington DC, 309–317.

[30] Matese, A., Di Gennaro, S.F. (2021). Beyond the traditional NDVI index as a key factor to mainstream the use of UAV in precision viticulture. Scientific Reports, 11, 2721. https://doi.org/10.1038/s41598-021-81652-3

[31] McFeeters, S.K. (1996). The use of the Normalized Difference Water Index (NDWI) in the delineation of open water features. International Journal of Remote Sensing, 17(7), 1425–1432. https://doi.org/10.1080/01431169608948714

[32] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15(1), 1929–1958.

[33] Kingma, D.P., Ba, J. (2015). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations (ICLR).

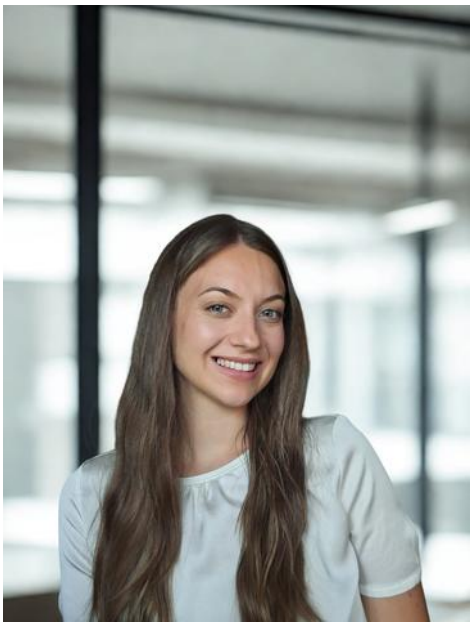
[34] Ioffe, S., Szegedy, C. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. International Conference on Machine Learning (ICML).

[35]Convolutional Neural Networks Explained. Encord. https://encord.com/blog/convolutional-neural-networks-explained/

[36] Skip Connections. AI Summer. https://theaisummer.com/skip-connections/

[37] Imad, M., Doukhi, O., Lee, D.J. (2021). Transfer Learning Based Semantic Segmentation for 3D Object Detection from Point Cloud. Sensors, 21(12), 3964. https://doi.org/10.3390/s21123964

Biography



Sonja Jamina was born on February 13, 1998, in Bijeljina, Republic of Srpska, Bosnia and Herzegovina. She completed her primary education at OŠ Vuk Karadžić in 2013 and graduated from Gimnazija Filip Višnjić in 2017. That same year, she enrolled at the Faculty of Sciences, University of Novi Sad, where she earned a bachelor's degree in Applied Mathematics - Technomathematics in 2021.

Following her undergraduate studies, she pursued a master's degree in Applied Mathematics - Data Science at the same university, completing all examinations with a GPA of 8.44 in 2022.

Since 2022, she has been working in the fields of artificial intelligence and data science, focusing on computer vision, Generative AI, agentic systems, and machine learning applications. She began her professional career at Vega IT Sourcing as a Data Scientist and Data Engineer before joining Publicis Sapient - Tremend as a Senior Data Scientist / AI Engineer, where she worked on enterprise AI and agentic systems. Since July 2026, she has been employed at EPAM Systems as a Senior AI Engineer.

Her professional interests include AI architecture and system design, Computer Vision, Agentic AI, and the application of machine learning technologies to real-world problems.

UNIVERZITET U NOVOM SADU

PRIRODNO-MATEMATIČKI FAKULTET

KLJUČNA DOKUMENTACIJSKA INFORMACIJA

Redni broj, **RBR**	
Identifikacioni broj, **IBR**	
Tip dokumentacije, **TD**	Monografska dokumentacija
Tip zapisa, **TZ**	Tekstualni štampani materijal
Vrsta rada, **VR**	Master rad
Autor, **AU**	Sonja Jamina
Mentor, **MN**	Dr Nikola Obrenović
Naslov rada, **NR**	Procena modela semantičke segmentacije zasnovanih na U-Net arhitekturi za detekciju redova vinograda na snimcima dobijenim bespilotnim letelicama
Jezik publikacije, **JP**	Engleski

Jezik izvoda, **JI**	Engleski
Zemlja publikovanja, **ZP**	Republika Srbija
Uže geografsko područje, **UGP**	Vojvodina
Godina, **GO**	2026.
Izdavač, **IZ**	Autorski reprint
Mesto i adresa, **MA**	Novi Sad, Prirodno-matematički fakultet, Departman za matematiku i informatiku, Trg Dositeja Obradovića 4
Fizički opis rada, **FO**	5 poglavlja / 43 strane / 37 lit. citata / 17 slika / 3 tabele
Naučna oblast, **NO**	Matematika
Naučna disciplina, **ND**	Primenjena matematika
Ključne reči, **KR**	UAV snimci, semantička segmentacija, U-Net, Dense U-Net, detekcija redova, vinograd, duboko učenje, precizna poljoprivreda, multispektralno snimanje, autonomna navigacija

Univerzalna decimalna klasifikacija, **UDK**	
Čuva se, **ČU**	Biblioteka Departmana za matematiku i informatiku, Prirodno-matematičkog fakulteta, u Novom Sadu
Važna napomena, **VN**	
Izvod, **IZ**	Ovaj master rad bavi se automatskom segmentacijom redova u vinogradu na osnovu multispektralnih ortomozaika dobijenih snimanjem dronom (UAV), kao osnovom za određivanje prohodne putanje autonomnog zemaljskog vozila (UGV) kroz prostor između redova - korak ka uvođenju autonomne navigacije u vekovnu poljoprivrednu tradiciju. Dve arhitekture dubokog učenja za segmentaciju, U-Net i Dense U-Net, trenirane su i upoređene na četiri kombinacije spektralnih kanala - RGB, RGB-NIR, RGB-RE i Green-RE-NIR - koristeći ručno digitalizovane poligone redova vinove loze kao ground truth. Performanse modela ocenjene su pomoću mera Intersection over Union, Dice koeficijenta, tačnosti po pikselu, preciznosti, odziva, F1-mere i specifičnosti. Utvrđeno je da je prisustvo bliskog infracrvenog (NIR) ili red-edge spektralnog kanala bitan faktor kvaliteta segmentacije: sve tri takve kombinacije postigle su Dice koeficijent iznad 0.87, dok je čisto RGB rešenje dalo merljivo slabije rezultate, a U-Net arhitektura je u svim testiranim konfiguracijama dostigla ili nadmašila kompleksniji Dense U-Net. Ovi nalazi pružaju konkretnu, na podacima zasnovanu osnovu za izbor senzora i arhitekture prilikom primene segmentacije redova u vinogradu kao dela stvarnog sistema za navigaciju UGV vozila, doprinoseći razvoju precizne vitikulture bez gubljenja iz vida tradicije na kojoj počiva.
Datum prihvatanja teme od	12.10.2026.

strane NN veca, **DP**	
Datum odbrane, **DO**	
Članovi komisije, **KO**	Predsednik: dr Davor Kumozec, docent Prirodno-matematičkog fakulteta u Novom Sadu Mentor: dr Nikola Obrenović, viši naučni saradnik Instituta BioSense u Novom Sadu Član: dr Lidija Fodor, docent Prirodno-matematičkog fakulteta u Novom Sadu

UNIVERSITY OF NOVISAD

FACULTY OF SCIENCES

KEY WORDS DOCUMENTATION

Accession number, **ANO**	
Identification number, **INO**	
Document type, **DT**	Monograph type
Type of record, **TR**	Printed text
Contents code, **CC**	Master’s thesis
Author, **AU**	Sonja Jamina
Mentor, **MN**	Nikola Obrenović, PhD
Title, **TL**	Assessing U-Net-Based Semantic Segmentation Models for Vineyard Row Detection in UAV Imagery
Language of text, **LT**	English
Language of abstract, **LA**	English

Country of publication, **CP**	Republic of Serbia
Locality of publication, **LP**	Vojvodina
Publication year, **PY**	2026.
Publisher, **PU**	Author’s reprint
Publ. place, **PP**	Novi Sad, Department of Mathematics and Informatics, Faculty of Science and Mathematics, University of Novi Sad, Trg Dositeja Obradovića 4
Physical description, **PD**	5 chapters / 43 pages / 37 references / 17 figures / 3 tables
Scientific field, **SF**	Mathematics
Scientific discipline, **SD**	Applied mathematics
Key words, **KW**	UAV imagery, semantic segmentation, U-Net, Dense U-Net, row detection, vineyard, deep learning, precision agriculture, multispectral imaging, autonomous navigation

Universal decimal classification, **UDC**	
Holding data, **HD**	The Library of the Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad
Note, **N**	
Abstract, **AB**	This thesis investigates the automatic segmentation of vineyard rows from UAV-acquired multispectral orthomosaic imagery, as a basis for inferring a drivable path for an unmanned ground vehicle (UGV) operating between the rows - a step toward bringing autonomous navigation into a centuries-old agricultural tradition. Two deep learning segmentation architectures, U-Net and Dense U-Net, were trained and evaluated across four spectral channel combinations: RGB, RGB-NIR, RGB-RE, and Green-RE-NIR, using manually digitized vine-row polygons as ground truth. Model performance was assessed using Intersection over Union, Dice coefficient, pixel accuracy, precision, recall, F1-score, and specificity. The availability of a near-infrared or red-edge spectral band was found to be the dominant factor in segmentation quality: all three such combinations achieved Dice coefficients above 0.87, while the purely visible-spectrum RGB combination performed measurably worse, and U-Net matched or outperformed the more complex Dense U-Net across every tested configuration. These findings provide a concrete, data-driven basis for selecting sensor and architecture configurations when deploying vineyard row segmentation as part of a real-world UGV navigation system, helping precision viticulture advance without losing sight of the tradition it is built on.
Accepted by the Scientific Board on, **ASB**	12.10.2026.

Defended on, **DE**	
Thesis defend board, **DB**	Chairperson: Davor Kumozec, PhD, Assistant Professor at the Faculty of Sciences, University of Novi Sad Mentor: Nikola Obrenović, PhD, Research Associate at the BioSense Institute in Novi Sad Member: Lidija Fodor, PhD, Assistant Professor at the Faculty of Sciences, University of Novi Sad