POLITECNICO DI TORINO

Master's Degree in Aerospace Engineering

Politecnico
di Torino

Artificial Intelligence for Convective Regions Segmentation in Radar Images

Candidate:
Thomas VOZZA

Supervisors:
Prof. Alessandro BATTAGLIA
Prof. Fabrizio STESINA

Contents

Abstract

The WIVERN (WInd VElocity Radar Nephoscope) mission, recommended by ACEO committee, aims to deliver global, three-dimensional observations of atmospheric winds, alongside measurements of reflectivity and brightness temperature linked to clouds and precipitation. A critical requirement for fully exploiting WIVERN data is distinguishing between convective and stratiform cloud regions. Convective areas, marked by strong vertical motions ($|w| > 1\text{m/s}$), are associated with intense precipitation and severe weather, while stratiform regions are broader with weaker vertical motion. This distinction is essential for improving numerical weather prediction (NWP), as WIVERN's horizontal wind measurements (VHLoS) are only valid in stratiform zones. This thesis proposes developing an Artificial Intelligence (AI) model for binary convective/stratiform classification using WIVERN observables such as vertical profiles of radar reflectivity and Doppler velocity (VLoS). The selected model is the U-Net neural network, widely used in biomedical image segmentation, here adapted for meteorological radar data. The network will be trained on synthetic datasets generated by WIVERN's end-to-end (E2E) simulator, which mimics radar observations based on high-resolution atmospheric models and accounts for radar physics, geometry, and satellite dynamics. This allows for large-scale, labelled training data without relying on costly field campaigns. The model will perform pixel-wise classification of simulated radar images, labelling each point as convective or stratiform based on the strength of vertical motions. The study will also explore the potential for further classification and analyze the influence of dataset characteristics on performance. Evaluation metrics will include Precision, Recall, F1-score, and Equitable Threat Score (ETS). The goal is to create a reliable tool to support WIVERN's scientific objectives and enhance global weather prediction.

1 Introduction

Earth observation from space plays a crucial role in the understanding of meteorological dynamics, especially in relation to the increasing frequency and intensity of extreme weather events. Recent decades have shown how global warming and changing climate patterns are influencing cloud formation, precipitation regimes, and storm behavior across multiple scales. As a result, there is a growing need for high-quality, global, and vertically resolved atmospheric observations to better support weather prediction, early warning systems, and climate research.

Within this context, the WIVERN mission (WInd VElocity Radar Nephoscope) emerges as one of the most promising satellite-based initiatives in Earth observation. Proposed as part of the European Space Agency's Earth Explorer 11 programme and recommended by the Advisory Committee for Earth Observation (ACEO), WIVERN is designed to address one of the most critical gaps in current atmospheric observation systems: the lack of direct, global measurements of wind profiles within clouds. The mission aims to provide high-resolution, three-dimensional measurements of line-of-sight winds, along with radar reflectivity profiles and brightness temperature values linked to clouds and precipitation.

A key requirement for fully leveraging WIVERN's potential is the ability to discriminate between stratiform and convective cloud regions. As will be discussed in detail in Chapter 2, this distinction is vital for the correct interpretation of Doppler velocity measurements, particularly for the derivation of horizontal wind components (VHLoS), which are only valid in stratiform areas where vertical motions are negligible. Convective regions, in contrast, are characterized by strong vertical motions and are typically associated with severe weather systems such as thunderstorms or tropical cyclones. Proper identification of these regions is therefore essential not only for the scientific exploitation of WIVERN data, but also for improving Numerical Weather Prediction (NWP) models.

Traditional approaches to convective/stratiform classification often rely on empirical thresholds applied to radar reflectivity fields or on features observed in previous satellite missions (e.g., TRMM, GPM). However, these methods can be limited in flexibility, and their accuracy may degrade in complex atmospheric conditions or with instruments operating at different frequencies or viewing geometries. The growing availability of synthetic data from satellite simulators, together with advances in Artificial Intelligence (AI), opens up new possibilities for automated and scalable classification methods based on data-driven learning.

In this thesis, we propose the development of a deep learning model to automatically classify WIVERN-like radar observations into convective or stratiform regions, using the observable fields of radar reflectivity and Doppler velocity as input. The selected model architecture is the U-Net, a convolutional neural network originally developed for biomedical image segmentation, which has demonstrated excellent performance in geophysical and remote sensing applications. U-Net's

encoder–decoder structure, combined with skip connections, allows the network to simultaneously capture fine spatial details and global context—making it particularly well-suited for pixel-wise segmentation tasks on radar images.

The model is trained on synthetic datasets generated using WIVERN's end-to-end (E2E) simulator, which realistically replicates radar measurements based on high-resolution atmospheric models such as WRF or SAM. These simulations account for radar physics, satellite orbit, scanning geometry, and signal degradation effects, allowing the creation of a diverse and representative training dataset without the need for costly field campaigns or yet-unavailable in-orbit observations.

The classification task is explored in three different configurations:

- Binary classification, distinguishing between stratiform and convective regions;
- Classification using continuous probability masks to describe transition regions;
- Multi-class classification, aimed at recognizing different intensities of convective activity.

Each scenario is addressed through careful preprocessing of input and target data, architecture tuning, and evaluation of the learning performance. The model's performance is assessed using standard classification metrics, such as Precision, Recall, F1-score, and the Equitable Threat Score (ETS), with particular attention to the model's ability to generalize to unseen simulated scenes.

Ultimately, the objective of this work is to contribute to the development of an AI-based tool capable of supporting the scientific and operational goals of the WIVERN mission. By enabling accurate and automated segmentation of convective structures in radar images, this research aims to enhance the reliability of global wind retrievals, improve weather forecast systems, and advance the use of AI in the next generation of satellite meteorology.

2 WIVERN Mission Context

Earth observation performed by satellites plays a crucial role in studying and understanding the meteorological phenomena of our planet. Satellites provide a unique, comprehensive perspective that enables the monitoring and, most importantly, the study of certain phenomena. The global warming observed in recent years has led to consequences across multiple aspects that cannot be overlooked. Changes in cloud configurations and precipitation patterns have been noted, which are reflected in the increasing frequency of extreme meteorological events. These atmospheric changes introduce new dynamics that must be studied to improve prediction capabilities and, consequently, better manage their impacts.
It is essential to emphasize the importance of accurate weather forecasts, both at the local level and for large-scale macro-regions. The increase in extreme weather events cannot be ignored, and accurate meteorological predictions not only help manage emergencies in terms of public safety but also mitigate economic losses and damages [1].
The instruments used in this field are continuously evolving, and, as a result, so are the data and information they require. For instance, the World Meteorological Organization (WMO) has identified a lack of globally distributed direct wind observations, which poses a limitation to the WMO's observation system. The data obtained from such observations are crucial for Numerical Weather Prediction (NWP) models, which require parameters related to wind, clouds, and precipitation to accurately represent the current atmospheric state, upon which the model bases its weather forecast [2]. Furthermore, the use of more advanced climate models demands new observations with higher spatial resolution (approximately 1 km) to fully leverage the increased complexity of these next-generation models [3].

2.1 Objectives and Measurements of the WIVERN Mission

The WIVERN (Wind VElocity Radar Nephoscope) mission, recommended by ACEO committee, will help bridge the gaps in current atmospheric observations by providing a range of valuable data and information for Earth observation.

- One of the key products of the WIVERN mission is the measurement of winds within cloud formations. This will help fill the gaps in the WMO observation systems, which require such data for studying the dynamics of certain events, including extreme ones, as well as for their prediction. Numerical Weather Prediction (NWP) models will also be able to use this missing wind information in stratiform cloudy regions to improve forecast accuracy. The precision of these forecasts depends not only on the most recent observations—provided in real-time by WIVERN—but also on having

as much information as possible about the current state of the atmosphere. It is evident that vertically resolved wind measurements within clouds on a global scale are highly valuable, significantly enhancing the accuracy of NWP model predictions [4]. These data were previously unavailable but will now be accessible thanks to the WIVERN mission, which will provide wind information within cloud formations, including those associated with cyclones and tropical storms, for the first time. The conically scanning radar onboard WIVERN measures the Doppler shift of water particles in the atmosphere, allowing for the calculation of wind speeds within cloud formations and in regions affected by extreme weather events.

- Another product of the WIVERN mission are the reflectivity profiles and brightness temperature values, which will be provided with higher spatial and temporal resolution. This informations are useful for studying the microscopic properties of clouds and precipitation. The observations are in real time and provide an excellent basis for climate models and NWP models, which, thanks to this data, improve their ability to predict and simulate precipitation and the evolution of cloud systems [4]. The increase in the spatial resolution of these models, which is expected to reach 1 km, makes it necessary to have this new data at the same resolution for validation [3].

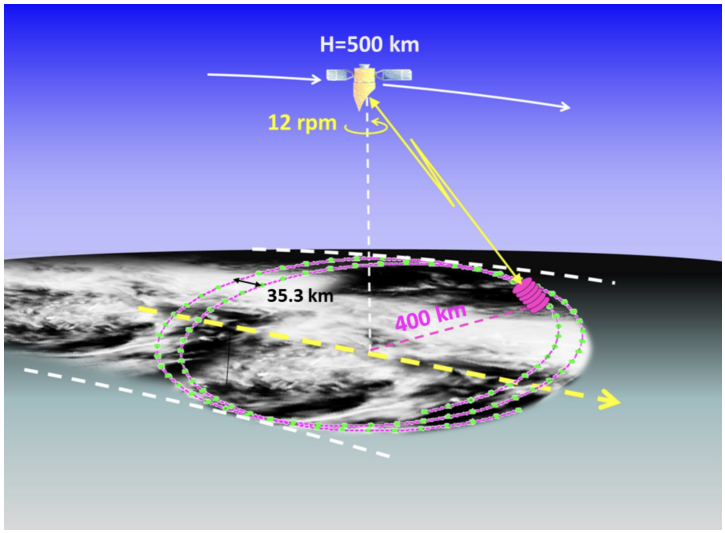


Figure 1: The figure shows a qualitative diagram of the WIVERN mission [5].

WIVERN uses a conically scanning dual-polarization Doppler radar at 94 GHz to measure line-of-sight winds (via the Doppler shift) and reflectivity profiles with a resolution of 640 m. Dual polarization helps prevent the decorrelation of the return signal phases caused by the satellite's motion. The polarization is orthogonal, allowing the two signals, emitted within a very short time interval ($T_{hv} = 20\,\mu s$),

to propagate independently. This approach enables more precise measurements, as the signals do not interfere with each other, allowing for much shorter time intervals between emissions and maintaining phase correlation for a more accurate Doppler velocity estimation [6]. The radar antenna has a 800 km wide swath that allows a revisit time of 1.5 days at the equator and 1 day over 50^o latitude [3].

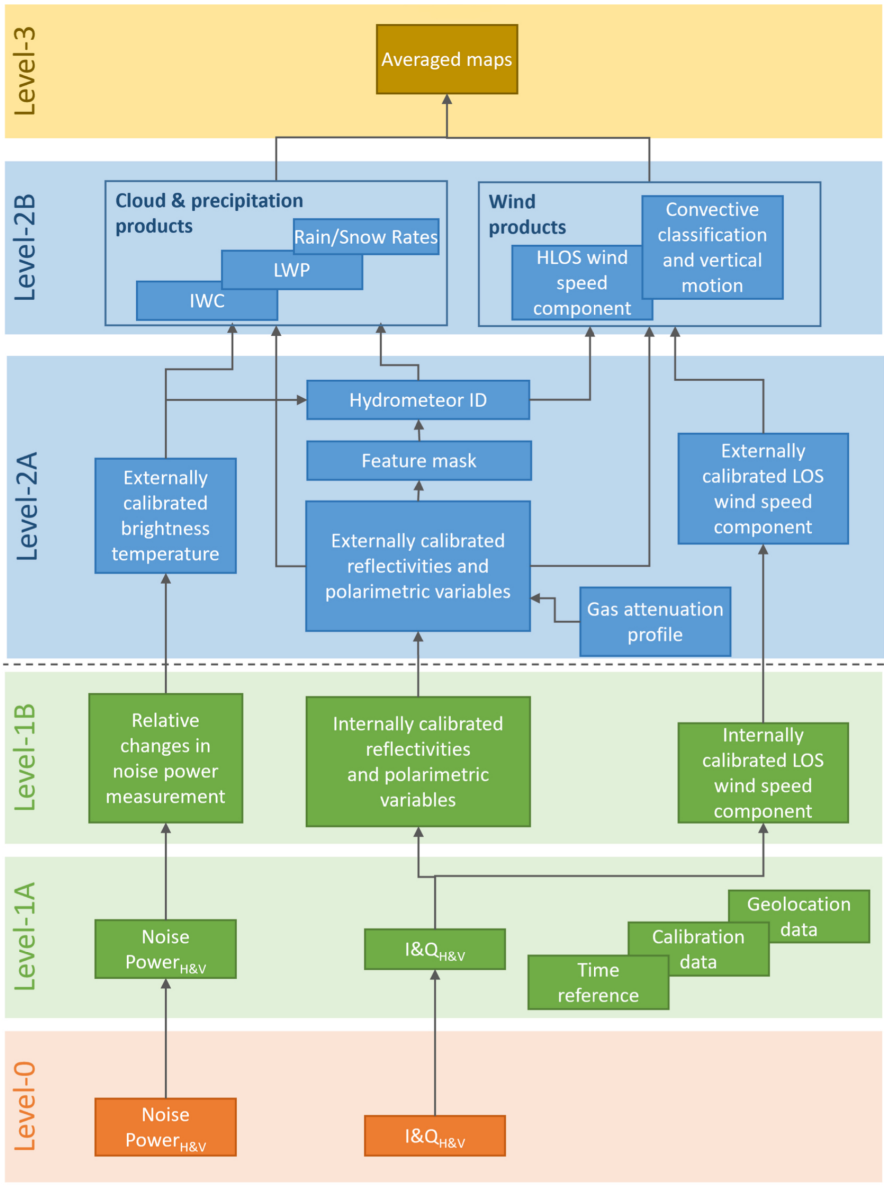


Figure 2: This scheme represents the various products derived from the measurements of the WIVERN mission. At "Level-0" are the raw instrument measurements, progressing to "Level-3" where averaged maps are produced. The classification work is positioned at "Level-2B", just before the final product levels. From the diagram, it is clear that the data used are the calibrated Line-Of-Sight (LOS) wind speed component and the calibrated reflectivity, available at "Level-2A" [4].

2.2 Convective and Stratiform Regions in Weather Dynamics

Convective regions are of great importance for the study of atmospheric dynamics, with consequences for meteorological phenomena. Convective clouds are responsible for the transport of large air masses to the upper regions of the troposphere [7] and are linked to regional climate changes, which in turn affect short-term seasonal forecasts [8]. The intensity and strength of certain storms and meteorological events are related to upward movement and transport: the greater this phenomenon, the larger the quantities carried to higher regions, increasing the likelihood of particularly extreme and severe weather events. Knowing this, it is crucial that such dynamics are well represented in prediction models. However, these models struggle to accurately estimate convective characteristics in the atmosphere. The lack of observations limits the progress of Numerical Weather Prediction (NWP) models, representing a gap that the WIVERN mission aims to fill. Distinguishing convective cloud regions from stratiform ones also provides valuable information. Winds in stratiform cloud regions have proven to be highly important, allowing for improved forecasting of severe weather events through NWP models and the global observation system. Effectively distinguishing these regions based on the primary collected data is one of the key aspects to develop.

In this context, this work involves developing an artificial intelligence model capable of using data obtained through WIVERN observations — namely reflectivity and Doppler velocity measurements - to distinguish between convective and stratiform regions in the observed cloud formations. The model will need to be trained to perform this classification based on a dataset that enables this analysis to be carried out as effectively as possible

2.3 Background of AI and Radar Images

In recent years, various approaches have been developed to perform classification or distinction in the atmosphere of precipitation or other meteorological phenomena. The classical methods employed are those based on low-frequency radars (TRMM, GPM) used on orbiting satellites. These are capable of identifying precipitation in stratiform regions as a bright band, i.e., a strong horizontal signal at around 4–5 km altitude. At this height, snowflakes melt and become coated in a layer of water, which reflects the radar signal [9]. For precipitation in convective areas, on the other hand, strong reflectivity signals are observed extending to high altitudes without showing a transition region where the phase change occurs [10]. High-frequency radars, such as the one used by WIVERN, present certain limitations. Despite their greater sensitivity, the signal becomes saturated and unreadable during heavy rainfall, as the water attenuates it [11] and causes multiple scattering [12], which distorts it. In such cases, it has been decided to observe the tops of clouds in order to detect overshooting of particles rapidly ascending into

the atmosphere, which indicates the presence of convection. These approaches are based on vertical reflectivity profiles and therefore do not take into account information about the velocity of precipitation.

There are also other approaches to classification using ground-based radar through the use of various algorithms. Threshold-based methods or simple pixel-wise instructions on measured field images have been employed. An example is [13], which implements classification based on the presence of a reflectivity peak and the contrast between it and its surroundings. There are also fuzzy logic-based methods, where instead of a hard classification, an uncertainty value is assigned to a point or pixel of interest [14].

The application of machine learning has represented a further step forward, leading to an increase in the accuracy of the results produced. Rather than applying rules and logic to the measured data fields, mathematical models such as neural networks are trained to autonomously learn to recognise and classify image pixels after being exposed to a number of already labelled examples. For instance, a CNN [15] can be used to perform multi-class classification based on the observation of overshooting from the tops of clouds.

The use of the U-Net architecture has become standard practice for tasks involving image segmentation and classification. It is widely used in the biomedical field, but it has also found application in geophysical contexts such as surface observation through multispectral imagery [16] or classification of such imagery [17]. From the perspective of atmospheric studies, U-Net has been used in several works. Its performance has been demonstrated in tasks such as forecasting downpours within 30 minutes of observation [18], or in the recognition and identification of convective cold pools in the atmosphere [19]. It has also been possible to generate hurricane risk maps by applying U-Net to fields produced by forecasting models [20].

3 Introduction to Artificial Intelligence and Machine Learning

Artificial Intelligence (**AI**) refers to that branch of science which studies how to use computers to imitate human behaviour and its characteristics—such as decision-making processes that allow problems of varying complexity to be solved with minimal or no human intervention. The goal of AI is to build systems capable of solving problems that may match or, in some cases, surpass human performance in specific tasks. Therefore, AI must evolve to address a variety of problem types, each requiring different techniques and tools which will define the machine's learning, perception, communication, and, more simply, reasoning processes.
The main challenge lies in the difficulty of replicating and programming what we humans refer to as *tacit knowledge* - that is, intuition or the implicit understanding we employ unconsciously when carrying out more complex tasks. For relatively simple tasks, it is possible to define explicit rules that help the machine identify and solve the problem. However, for more complex tasks, setting parameters and rules becomes significantly more challenging, and tacit knowledge becomes nearly impossible to define and program into a computer.

Machine Learning (**ML**) is a subfield of Artificial Intelligence that enables the replication of this more advanced aspect. Instead of providing the computer with precise rules and describing how to solve a specific problem, the computer learns autonomously - using the data and information provided, and through experience - how to solve it and even improve its performance over time (in a manner similar to human learning). In practice, this is achieved by using algorithms that learn iteratively from training data contained in a dataset. These algorithms enable the machine to identify hidden or more complex patterns with the aim of building analytical models capable of making repeatable and accurate decisions.

Within the field of Machine Learning, three main types of learning approaches can be distinguished:

- **Supervised Learning** requires a training dataset that contains both input data and corresponding output responses (targets). Training is performed using solved examples contained in the dataset. During the learning phase, the model's parameters are appropriately adjusted based on these input–target pairs. Training is considered complete once the model is able to predict the target for a new, previously unseen input. We distinguish between regression problems, where a numerical value is predicted, and classification problems, where the model must assign the correct class label.

- **Unsupervised Learning** occurs when the model is asked to identify patterns and features within a dataset that contains only input data - without targets or labels, in contrast to supervised learning.

- Unlike supervised and unsupervised learning, **reinforcement learning** does not rely on a static dataset. Instead, data is generated dynamically through interaction with the environment subject to certain constraints that produce effects, thereby altering the system's state. If an action leads to a positive result for the system, it is rewarded with a score (reward). The model's training in this scenario leads it to independently find the best strategy to maximise the reward (and minimise error), ultimately solving the given task.

Machine Learning still requires data processing before the data can be provided to the model. The dataset must accurately represent the features that need to be extracted, and human intervention is often necessary to ensure this.

With **Deep Learning** (**DL**), some of these limitations are overcome by using more complex models which, although they require greater computational resources, are capable of extracting features directly from raw data without the need for preprocessing or any other human intervention on the input dataset.

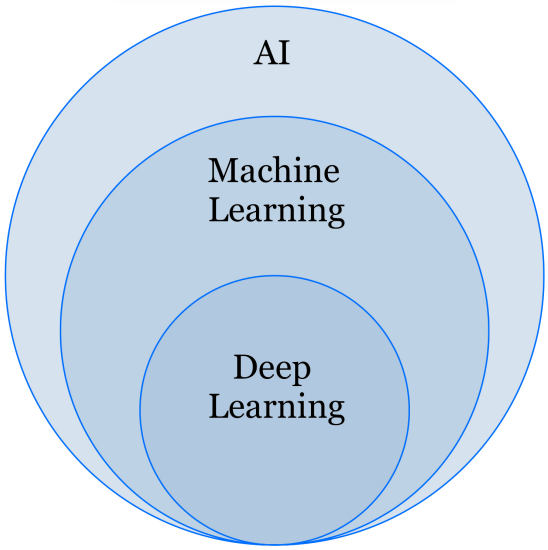


Figure 3

3.1 Deep Learning and Artificial Neural Network (ANN)

Deep Learning is a subcategory of Machine Learning. It developed as a result of the increased computational power of computers and the larger amounts of data that can be made available for training models. Unlike a Machine Learning model, a Deep Learning model is capable of working with raw data and extracting features from a dataset without the need for human intervention or manipulation. Models that use this type of learning have structures that are similar to our neurons, both functionally and sometimes even structurally.

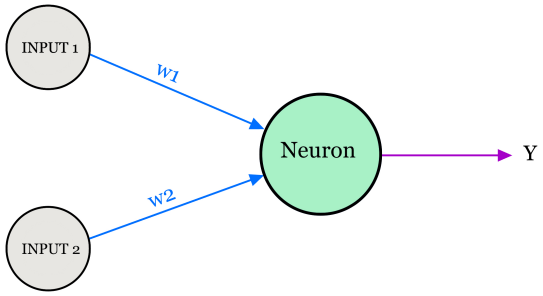


Figure 4: Neuron structure. It takes 2 input-which are weighted-and returns the output after an activation function.

In this context, there are Artificial Neural Networks (ANNs). These models are based on the biological neural model, which is replicated as a mathematical model and forms the foundation of more complex models and networks used in deep learning. The smallest unit that makes up a neural network is the neuron itself. It takes in input data, and through a function with weights, it can activate and produce an output. Multiple neurons make up what is called a layer in the neural network. An ANN can consist of several layers, with each one receiving information and producing an output. The initial layers take in the data and extract the simplest, low-level features. As the numbers of layers increase and the architecture of the network becomes more complex, it becomes possible to extract more abstract features: in this case, we refer to Deep Neural Networks (DNNs). The learning process can occur through backpropagation: based on the output provided by the network and the target given in the training dataset (with supervised training), the model's weights can be updated at each iteration using backpropagation combined with an optimization algorithm such as stochastic gradient descent (SGD) or Adam.

3.2 Convolutional Neural Network (CNN)

A **Convolutional Neural Network** (**CNN**) is a deep neural network architecture specialised in image analysis, but it is also employed in other fields such as computer vision, speech processing, and face recognition. The structure of a CNN is based on the perceptron model, but this time with multiple layers arranged in succession. In the case of image analysis, the image enters the network through an input layer, then passes through the hidden layers, and finally produces a result via an output layer. Each individual layer forms the basic building block upon which the CNN structure is based. Unlike a perceptron, each layer in a CNN performs various operations on the input image.

- **Convolution** is perhaps the most important step carried out. It involves applying filters to the image, which are understood as smaller matrices of weights (w), known as *kernels*. These matrices move across the image both vertically and horizontally. At each position of the filter over the image,

a sum is calculated by multiplying the pixel values of the image with the corresponding weights of the overlapping filter. This produces a new image called a feature map, which, depending on the filter used (i.e., how the weights in the kernel are chosen), highlights certain characteristics present in the image. The application of these filters for performing convolution can be modified using specific parameters. *Stride* defines how many pixels the kernel moves across the image (increasing the stride reduces the size of the resulting feature map). *Padding* refers to the value of pixels added around the image so that the kernel can still interact with the border pixels during convolution.

- **Pooling** is an operation that reduces the size of the feature maps and is performed after convolution. The goal is to generalise the dominant information extracted during the convolution phase. Pooling involves a kernel that moves across the feature map and extracts a value using an operation (e.g., taking the maximum or calculating the average) from the pixels covered by the kernel at each position.

- **Activation functions** are what allow this type of architecture to be so accurate. The non-linearity of these functions enables the model to learn more complex features that a linear combination of nodes could never capture (in the absence of non-linearity). Common activation functions include **SIGMOID**, which returns a value between 0 and 1; and **ReLU**, the most commonly used activation function in CNNs, which offers low computational cost.

The structure of a CNN consists of a sequence of layers where these operations are carried out. Different network architectures can be created based on the number and arrangement of these layers.

- The **Input layer** consists of nodes that take the image as input to the CNN and pass it to the nodes in the subsequent layers.

- The **Convolution** and **Pooling layers** are where the above operations are performed. The nodes in these layers pass information to the next layer through activation functions.

- The **Flatten layer** may be used to convert a sequence of two-dimensional nodes (corresponding to the feature maps from previous layers) into a one-dimensional configuration, if required by the output layer.

- A **Fully Connected Layer** (**FC Layer**) connects every node in the current layer to every node in the following layer.

The parameters of the network that must be learned during training are the weights (w) and the biases (b). In convolutional layers, the weights are the values

in the filter matrices (kernels) that slide across the image and help extract the relevant features. In fully connected layers, each connection between nodes has an associated weight and bias that allow the model to map the extracted features to output classes. Biases are values added to the weighted sum of inputs and allow neurons to activate even when the input is zero. There is one bias per kernel and per node connection in FC layers. An important aspect of CNNs is weight sharing in convolutional layers. The weights, defined by the filter matrix, are shared across the entire image; thus, the number of parameters depends not on the image size but on the size of the filter and the number of filters used per layer. This significantly reduces computational complexity compared to FC layers, where weights are defined for every single connection between nodes, leading to a rapid increase in the number of parameters as image size grows. Unlike a standard neural network, a CNN is not affected by translation. Therefore, the recognition or segmentation of a subject does not depend on its position within the image. This is due to the fact that weights and biases are shared across all neurons in a given layer.

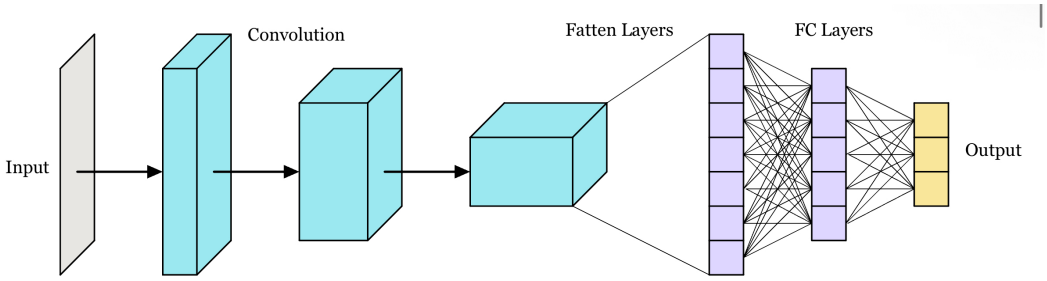


Figure 5: CNN architecture. Blue: convolutional layers. Purple: Fully connected layers.

3.3 U-Net for Image Segmentation

U-Net networks are a convolutional neural network architecture developed primarily for image segmentation, where each pixel in the image is assigned to a specific class (for example, identifying object boundaries or distinguishing specific areas within the image). Originally introduced for biomedical image segmentation, U-Net has demonstrated great versatility and has spread to many other fields, including radar image analysis. Its name "U-Net" derives from its characteristic U-shaped architecture, which effectively balances the capture of global context and the preservation of spatial details. The structure of U-Net can be conceptually divided into two main paths: a contracting path (or encoder) and an expanding path (or decoder)

- The contracting path (**Encoder**) is similar to a typical CNN used for classification and serves to extract features present in the input image and

progressively acquire higher-level contextual information. The encoder consists of multiple levels, each comprising a convolutional layer followed by a pooling layer (typically max-pooling). This reduces the spatial dimension of the image while simultaneously increasing the number of feature maps obtained through convolution. The number of levels in the encoder branch (which is then mirrored in the decoder) defines the network's complexity and its ability to extract increasingly abstract features from the images. As the spatial resolution decreases, the feature maps contain more abstract and position-invariant information.

- The expanding path (**Decoder**) is responsible for reconstructing the segmentation of the image at the same resolution as the input image. The flow begins at the bottom of the U with the low-resolution feature maps generated by the encoder. These are "decoded" through a series of upsampling operations (such as deconvolution or transposed convolution) and convolutions. Upsampling operations, particularly **transposed convolution** are crucial in the expanding path. Unlike standard convolution that reduce spatial dimensions, transposed convolution operates "in reverse," applying filters to expand the spatial dimensions of the input. This process is designed to learn a transformation that reconstructs the spatial resolution lost during pooling. Practically, transposed convolution distributes the values of low-resolution feature maps over a larger area, thereby creating a higher-resolution feature map. This is vital for transitioning from abstract, low-resolution features back to a detailed, full-resolution representation required for pixel-level segmentation.

- A crucial aspect of U-Net lies in the **skip connections**, which link corresponding feature maps in the contracting path of the encoder with those in the expanding path of the decoder. The skip connections connect feature maps that have the same spatial resolution but originated from different stages of the network (encoder). The fusion of these maps typically occurs via concatenation. This means that the features learned by the encoder (which contain fine spatial information) are "stitched" together with the decoded features from upsampling (which contain high-level contextual information). This allows the decoder to combine semantic richness (the "what" of an object) with spatial precision (the "where" it is located and its exact boundaries), significantly improving the quality of the final segmentation. At the end of the expanding path, a final convolution and an activation function (such as sigmoid for binary segmentation or softmax for multi-class segmentation) produce the final segmentation map, which matches the input image in size.

Compared to a traditional CNN used for classification, U-Net offers significant advantages for segmentation tasks. U-Net produces a prediction for each individual pixel, making it ideal for tasks that require pixel-level understanding. The skip

connections are the key to the model, allowing the network to combine high-level information (semantic context) with low-level information (spatial details) — a crucial requirement for accurate segmentation.

Furthermore, U-Net is known for its ability to achieve good results even with relatively small training datasets, thanks to its efficient architecture that extensively leverages context expansion through encoding and the recovery of details through decoding and skip connections.

However, its main drawback compared to simpler architectures lies in its greater computational complexity, especially during the expansion phase, and in the need for more substantial hardware resources for training, due to the large number of parameters and the multiple convolutional and upsampling operations involved.

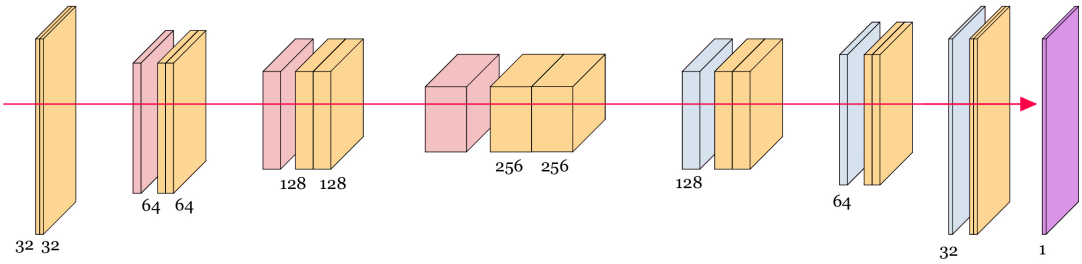


Figure 6: U-Net architecture. Yellow: convolutional layers. Red: pooling layers. Blue: unpooling layers. Purple: output

3.4 Training the Network

During the training of a neural network, the main objective is to optimise the network's weights and biases so that it can produce predictions that are as precise and accurate as possible. This process is driven by various factors and parameters, which must be carefully selected in order to ensure a successful outcome.

One of the fundamental elements involved is the **loss function**, which quantifies the discrepancy between the model's predictions and the expected values (labels). In classification problems, such as the one under consideration, commonly used loss functions include cross-entropy, which measures the distance between the predicted probability distributions and the actual ones. In regression tasks, on the other hand, functions such as the mean squared error (MSE) or mean absolute error (MAE) are typically employed, as they provide a numerical indication of the model's prediction error. Minimising the loss function thus corresponds to reducing the model's error and consequently improving its predictive performance.

Loss minimisation is achieved through an iterative process in which, at each training cycle, the input is propagated through the layers of the network (forward pass) to generate an output. The resulting error is then propagated backward through

the network (backpropagation) to update the weights. **Backpropagation**, or error backpropagation, is the fundamental algorithm that enables the efficient computation of the gradients of the loss function with respect to each of the network's parameters. After the forward pass, the loss is computed by comparing the network's output with the true label. This error is then propagated backwards through the layers using the chain rule of differential calculus, yielding the partial derivatives of the loss with respect to each weight. At each layer, these gradients indicate how much each weight contributed to the final error. With this information, the weights are updated in a way that progressively reduces the loss. The parameter update follows the rule:

$$w_{\text{new}} = w_{\text{old}} - \eta \frac{\partial L}{\partial w} \tag{1}$$

where η is the learning rate, L is the loss function, and $\partial L/\partial w$ is the gradient of the loss with respect to the weight, computed via backpropagation. This formula represents the basic principle of weight update in gradient-based optimization, where parameters are adjusted in the direction that reduces the loss. In practice, more sophisticated optimization algorithms are often employed—such as Adam, RMSprop, or AdaGrad—which extend this basic rule by introducing mechanisms like adaptive learning rates and momentum. Despite these differences, the underlying idea remains the same: weights are updated using gradient information to iteratively minimize the loss function. This mechanism is particularly efficient due to the layered structure of neural networks, which allows the reuse of intermediate gradients computed in deeper layers to efficiently calculate those in earlier layers.

The behaviour of the training process can be significantly influenced by a set of hyperparameters, which are not learned by the network but must be set in advance:

- The **learning rate** is among the most critical: it determines the magnitude of the weight updates. If set too high, training may become unstable or diverge; if too low, it may converge extremely slowly or get stuck in local minima.
- The **batch size** defines how many samples are processed simultaneously for each weight update. Small batch sizes produce noisier gradients, which can enhance generalisation; larger batches offer more stable gradient estimates but require more memory and may lead to less flexible solutions.
- The **number of epochs** refers to how many times the entire dataset is processed by the network. Increasing this number can improve performance, but beyond a certain point it can lead to overfitting.

Optimisation algorithms are responsible for updating the weights and directly influence the learning efficiency. The most basic is gradient descent, which updates weights proportionally to the gradient of the loss function. More advanced

variants, such as Stochastic Gradient Descent (SGD), use a random subset (mini-batch) of the dataset for each update, speeding up training and introducing variability that can help escape local minima. More sophisticated optimisers like Adam (Adaptive Moment Estimation) combine the speed of SGD with an adaptive adjustment of the learning rate for each parameter, using estimates of the gradient's first and second moments (mean and variance), making it particularly effective for deep networks.

Another crucial aspect is managing **overfitting**, which occurs when the model learns not only useful patterns from the data but also noise and dataset-specific irregularities, thus performing poorly on unseen data. Several techniques can be employed to mitigate this issue: *regularisation* (L1 or L2, also known as weight decay), which penalises excessively large weights; *dropout*, which randomly disables a portion of neurons during training, preventing the network from becoming overly reliant on specific internal pathways; and early stopping, which halts training once performance on the validation set begins to degrade. An additional approach is *data augmentation*, which increases dataset variety by generating synthetic examples through transformations such as rotations, translations, noise addition, and zooming, thereby encouraging the network to learn more robust representations. Finally, *batch normalisation*, which normalises neuron activations within each batch, can improve training stability and speed, while also contributing to better generalisation.

The interaction among all these components – loss function, optimisation algorithm, backpropagation, hyperparameters, and regularisation techniques – is fundamental in determining the effectiveness of the training process and the final quality of the model.

4 Dataset for ML Model Training

Creating a dataset with the right information that best represents the characteristics we aim to recognise is a fundamental aspect—if not the most important one—in developing and training a solid and accurate AI model. For this reason, it can also be one of the main challenges to tackle and overcome. It is therefore crucial to describe the type of data being used and how it is generated, as this helps define how to effectively proceed with the analysis we want the model to learn to perform.

In this context, the network we intend to train will work with data related to velocity and reflectivity fields, which WIVERN will eventually be able to produce. Additional information will also be available for the network to learn from and will be employed in the training process. All of this data will be provided by an end-to-end (E2E) simulator, which allows for the generation of image and data datasets with relative ease, without the need for in-situ observation campaigns.

While such campaigns could be effective [21], they undoubtedly require a greater effort on multiple fronts and might not meet the demands for the quantity of data and information that a model would need to have at its disposal. In this regard, the simulator can generate an image and data dataset of any necessary size, with the only limitation being the capacity of the model handling it and, consequently, the computational cost associated with the training phase.

4.1 E2E Simutaor

The simulator is a software tool developed to evaluate the performance of WIVERN's Doppler radar prior to its deployment. The simulator replicates the entire process of data acquisition and measurement that the radar will need to perform, starting from high-resolution cloud models that recreate the atmospheric scenes to be observed, all the way to the generation of final data, also taking into account the errors that may affect the measurements [5]. Below are the models used by the simulator to reproduce the measurements.

- **Atmospheric Model**. The atmospheric model is what enables the generation of the atmospheric scenes targeted by the radar observations. It is based on the use of global circulation models (ECMWF) and high-resolution models such as the System of Atmospheric Modelling (SAM). The latter, in particular, is capable of explicitly representing turbulence within cloud structures in the atmosphere. It can generate three-dimensional wind profiles, hydrometeor distributions, and other profiles of atmospheric variables such as temperature and relative humidity. These profiles are then interpolated to match the resolution of the WIVERN radar. The high resolution of the model allows testing the performance of the WIVERN mission radar

in measuring Doppler velocity and reflectivity under conditions such as convective storms, stratiform clouds, and tropical cyclones [5].

- **Orbital Model and Scanning Geometry**.This model allows for the description of WIVERN's orbit around the Earth and the orientation of the radar. With this information, it is possible to determine which portion of the atmosphere is being observed and, based on the satellite's attitude, at what angle. Orbital data is provided by the model at every moment, making it possible to replicate the effect of the observation angle on the Doppler velocity and to optimize the mission design in terms of observation coverage [5].

Parameter	**Value**
Satellite altitude, h_{sat}	500 km
Satellite velocity, v_{sat}	7600 m/s
Off-nadir pointing angle	38^o
Incidence angle, i	41.6^o
Output frequency	94.05 GHz
Pulse width	3.3 μs
Antenna beamwidth (3 dB)	0.071^o
Circular antenna diameter	3 m
Rotation speed	12 rpm
Footprint speed	500 km/s
Transmit polarisation	H or V
Cross-polar isolation	$< 25\,dB$
Single pulse sensitivity	18 dBZ
H-V pair repetition frequency	4 kHz
Range sampling distance (rate)	100 m (1.5 MHz)
Number of H-V pairs per 1 km integration length	8

Table 1: WIVERN radar system parameters

- **W-band Scattering**. The model must compute, at each point, the scattering properties of the atmosphere caused by rain, snow, cloud droplets, or ice crystals. Each type of particle interacts with the signal in a different way, so the model takes into account how and in what quantities particles are distributed in the atmosphere in order to properly weigh their effects. Scattering properties are based on tabulated values that consider the characteristics of the particle distributions (such as diameter and also temperature). Electromagnetic interactions are modelled using Mie theory, which is accurate as long as the particles are spherical. The diameter distributions for rain and snow are exponential in nature [6].The effects of dichroic media — materials that behave differently depending on the signal's polarization — are not modelled. However, neglecting these aspects can lead to Doppler

measurement errors due to signal pulse decorrelation. To mitigate this error, the simulator uses statistical values of LDR (Linear Depolarization Ratio) derived from past observations at the Chilbolton Observatory. These are sufficient to estimate the impact of *ghosts*, secondary phenomena that cause subtle distortions in the measurements but do not result in significant errors [5].

- **Surface Model**. This model describes the behaviour of the radar signal reflected from the Earth's surface. The LDR value is used as a measure of how much the polarization of the radar signal changes after being reflected by the surface; this parameter is related to the surface roughness. The σ_0 value describes how much of the radar signal is reflected back to the sensor by the surface and is normalized by unit area. Coastal regions, being transitional zones, assume intermediate values between those of land and sea [5].

- **Point Target Response**. This model describes the radar's capability to detect and characterize point targets. It enables the assessment of the radar's spatial resolution and discretization performance. This evaluation is carried out through comparisons with experimental measurements on known targets or by applying transfer functions [5].

- **Antenna Pattern**. The antenna pattern is defined using a simple circular Gaussian. This model is important for describing how the radar signal illuminates the scene and how the reflected signals are received. The pattern includes a main lobe and, if necessary, side lobes that can be added to the modelling. Off-axis attenuations are also considered, as they affect the quality of the radar measurements [5].

$$G(\theta) = G_0 \cdot \exp\left(-4 \log 2 \cdot \left(\frac{\theta_a}{\theta_{3\,dB}}\right)^2\right) = G_0\, f_a(\theta_a) \tag{2}$$

- **Mispointing Modelling**. To measure and calculate wind velocity accurately, it is essential to know precisely where the radar is pointing, as small errors can lead to significant and non-negligible measurement inaccuracies [22]. In orbit, the elevation angle is easier to control and its error can be known with greater precision, unlike the azimuth angle. For this reason, azimuth-related errors are modelled using a Power Spectral Density (PSD), which describes how much energy the error contains at each frequency (low-frequency components are typically associated with thermal deformation of the antenna, while high-frequency components are linked to vibrations). The spectrum derived from the PSD allows the generation of a realistic time series of pointing errors using an Inverse Fast Fourier Transform (IFFT).

These errors are then incorporated into the wind measurement process. Despite modelling a realistic sequence of errors, the final measurement error remains small [5].

The radar determines velocity through the phase shift between two consecutively transmitted pulses. However, when the target velocity is too high, ambiguity can arise between signals, making them indistinguishable to the radar. This results in a maximum detectable velocity known as the Nyquist velocity, which depends on the radar wavelength and the time interval between pulses.

$$v_{Ny} = \pm \frac{\lambda}{4\ PRI} \tag{3}$$

To increase this limit, the Pulse Repetition Interval (PRI) can be reduced. However, this also decreases the maximum unambiguous range and reduces pulse-to-pulse correlation, which in turn affects measurement accuracy. By transmitting pulses with different polarisations at very short intervals (e.g., $\tau = 3.3\,\mu s$) and using a low repetition frequency ($4\,kHz$), it is possible to increase the Nyquist velocity while maintaining unambiguous detection.

The radar records time series of the I and Q components (in-phase and quadrature) [23], but the model provides level 1 observables, which are the Doppler velocity and the reflectivity. A method based on established theories is used to directly estimate the measurement uncertainty. Contributions from surface scattering (signal reflected by the ground or the ocean) and atmospheric volume scattering (such as clouds, rain or snow) are taken into account.

- **Reflectivity**. The signal received from the atmosphere is calculated by integrating over the entire atmospheric volume from which the radar return is received. The return N is expressed as a function of the water content in dBZ. The integration is performed over the solid angle of the radar beam to obtain the return at different distances. The radar power reflected from the surface is calculated by integrating over the surface in view of the radar, taking into account atmospheric attenuation, distance, and the radar equivalent cross-section of the surface (which depends on the type of terrain considered) [24]. This second contribution is always expressed in dBZ. The sum of these two contributions determines the total radar return.

  $$Z_{VV}(r) = Z_{VV}^{surf}(r) + Z_{VV}^{atm}(r) \tag{4}$$

  The different polarizations of the radar signals can create *ghost* signals due to the fact that part of the vertically polarized signal can end up in the channel of the other polarization and vice versa. These cross-polarization contributions are also simulated [5].

- **Doppler Velocity**. Doppler velocity is estimated by processing the pulse pairs generated by the radar. The phase measurements that lead to the

estimation of Doppler velocity are noisy. They can be influenced by electronic noise, signal width (Doppler spread), and unwanted returns (*ghosts*). A normally distributed Gaussian random noise is generated and then added to the simulated velocity. If the obtained velocity exceeds the Nyquist limit, it is folded back into its range (aliasing phenomenon) [5].

4.2 Dataset features

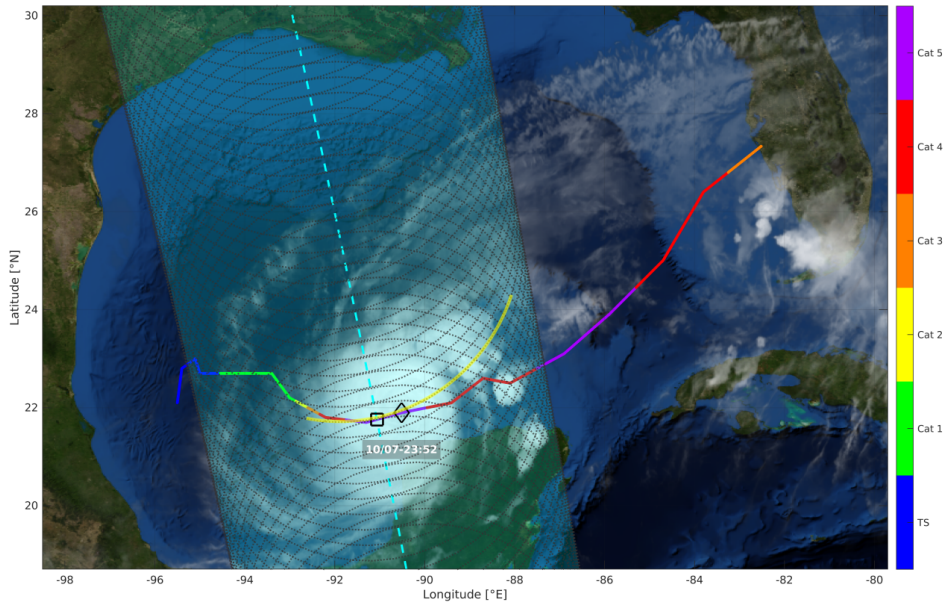


Figure 7: Representation of the Hurricane Milton scene, which will be the subject of WIVERN measurement simulations via the E2E simulator. The hurricane's trajectory is depicted, alongside the categories reached during its evolution. The WIVERN ground track is also shown, with the corresponding passes of the radar's conical scanning.

The observables used for this study were obtained through measurements simulated using the end-to-end WIVERN simulator. Specifically, the WRF (Weather Research and Forecasting) model was used to simulate the evolution of Hurricane Milton, which serves as the case study. The event was observed from 6 October 2024 at 10:00 UTC to 8 October 2024 at 00:00 UTC at hourly intervals, for a total of 39 hours during which it evolved from a tropical storm into a category 5 hurricane. Each simulated scene spans 1250x1250x20 km^3, with a horizontal resolution of 1 km and a vertical resolution of 70 m. For each of these scenes, WIVERN measurements were simulated for intervals of 200 seconds, corresponding to 40 radar rotations. Simulations included 13 satellite ground tracks between -6° and 6° longitude, spaced at 1° intervals. The available dataset covers a total of 5.5 million kilometres along the satellite's ground track. This was divided into 500

km segments, resulting in 10,912 images saved in NetCDF4 format using 32-bit floating-point precision. It is important to note that convective regions represent a minority in the dataset, as is typical in atmospheric systems. Areas with strong convection ($w > 3\ m/s$) account for just 0.62 % of the dataset, while those with moderate convection ($1\ m/s < w < 3\ m/s$) make up 3.67 %. Convective regions are under-represented and constitute a minority. From the simulations of WIVERN measurements and observations, the three variables used for training the network are obtained.

- One of WIVERN's products is the **Doppler velocity**, which represents the velocity component of particles along the radar's line of sight. It has been observed that large variability in Doppler velocity indicates the presence of convection [40]. Despite some attenuation, the WIVERN radar at 94 GHz is still sensitive to these variations in $VLoS$.

- **Radar reflectivity** is another WIVERN observable available for network training. High reflectivity values (in particular above 10 dBZ and beyond 10 km altitude) indicate the presence of large precipitation particles, such as rain or hail, which are lifted by intense updraughts typical of deep convection [25][26].

- **Vertical velocity** is not a direct WIVERN product; it is computed through simulation and corresponds to the projection along the line of sight of the true vertical velocity component ($wLoS$). To obtain the actual vertical component, this is divided by the sine of the line-of-sight incidence angle. This is the measurement used to construct the target images employed for model training. Depending on the processing applied, it is possible to build various types of maps and address multiple case studies.

5 Methodology

The aim of this work is to develop a neural network model that can be used for the segmentation and classification of convective and stratiform regions, based on measurement products derived from WIVERN observations. The simplest approach taken in this work involves binary segmentation of the two required classes. A further development of the network includes multi-class segmentation, which can be used both to differentiate between varying intensities of detected convection. Several approaches were used to address the problem, both from the perspective of pre-processing the dataset images and in terms of network architecture, employing models of varying size.

Below is the general methodology adopted for the various case studies, including the pre-processing of the input images, the approach chosen for training along with the related measures adopted, and an overview of the validation metrics selected to assess the results obtained with the model. The individual case studies addressed are then considered, presenting the processing of the targets that the network must learn to reproduce, any specific measures adopted for the training phase in relation to the case study in question, and the values of the metrics described earlier.

5.1 Model Architectures

The chosen network architecture, as previously mentioned, is the U-Net. Several variants of this architecture have been employed, differing only in their depth—that is, in the number of layers present in the encoder and decoder. The greater complexity of the deeper networks results in models that can achieve better performance, but also come with a higher computational cost during training due to the significant increase in the number of parameters to be computed. All architectures implement the same features, such as skip connections between corresponding encoder and decoder layers, bilinear up-sampling, and dropout at the bottleneck with a probability of 0.2 to prevent overfitting during training.

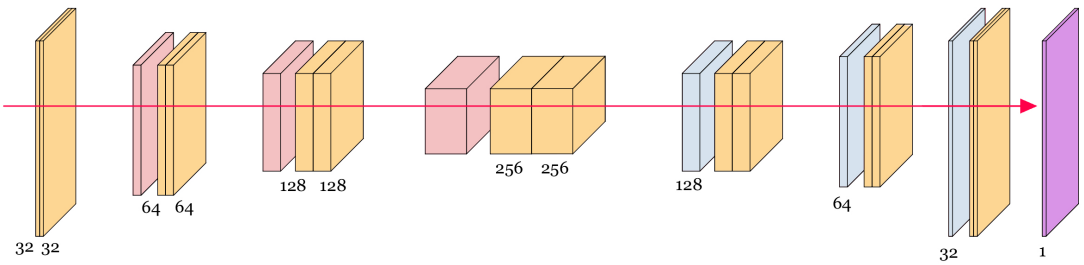


Figure 8: U-Net architecture (Small model). Yellow: convolutional layers. Red: pooling layers. Blue: unpooling layers. Purple: output. It is reported the number of channel for each layers.

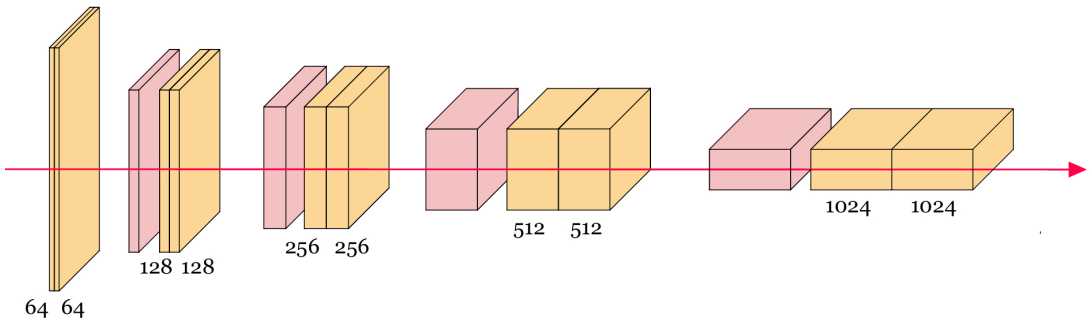


Figure 9: U-Net Encoder architecture (Larger model). Yellow: convolutional layers. Red: pooling layers. Blue: unpooling layers. Purple: output. It is reported the number of channel for each layers. It is reported the number of channel for each layers.

5.2 Training Setup

The entire dataset available was divided into two parts. The training dataset consists of 90% of the files and is used by the model to learn the features of the fields provided as input. The validation dataset consists of the remaining 10% of the total files, and the information it contains is not used for training the network. These images are used to compute the evaluation metrics for the models being trained and to calculate the loss using the loss function at the end of each training epoch. During training, the model with the lowest loss on the validation dataset is saved, along with the model obtained at the end of the final training epoch. A learning rate schedule has been implemented, consisting of a reduction in the learning rate by a factor γ after the loss on the validation dataset fails to improve for a specified number of consecutive epochs (patience). The specifications of the computer used for computation and training are reported in table 2

CPU	GPU
AMD Ryzen 7 9700x	NVIDIA 5070 Ti

Table 2: Computer specification

Multiple training iterations were carried out to obtain a better understanding of the influence of certain parameters. Several values were used for the loss function weight; the performance was observed when using only one of the two available inputs; and finally, the impact of the model's size on its performance was assessed. For each training run, both the final model at the end of training and the model that achieved the lowest loss on the validation dataset (calculated at each epoch) were saved. In general, the other training hyperparameters were kept unchanged across iterations, in order to highlight the effect of these specific modifications (Table 3).

Epochs	Batch Size	Learning Rate	Patience	LR Factor γ
100	16	0.001	5	0.5

Table 3: The Patience is the number of consecutive epochs without improvement in the validation loss that has to be waited before reducing the Learning Rate (LR) by the factor γ (Learning Rate Scheduler).

The training were carried out using different loss function based on the purpose of the case study.

- For binary segmentation the `BCEWithLogitsLoss` loss function was used. This function takes the logits output by the network (which are the values representing the probabilities of belonging to the target class) and applies the `Sigmoid` activation function, thereby generating a probability map with continuous values in the range between 0 and 1. The model weights are then updated based on the loss, which is calculated by comparing the prediction just obtained with the target from the dataset associated with the input. Given the imbalance of the classes represented in the dataset, the loss function is adjusted with a specific weight that makes the model more sensitive to the positive class, penalising false negatives to a greater extent. In the training process for the continuous target case study, the same loss function is used.

- For multi-class segmentation, the `CrossEntropyLoss` loss function was used. This function is suitable for problems in which each pixel can belong to one of several classes. The network outputs a tensor of logits for each class, and the `CrossEntropyLoss` internally applies the `Softmax` activation function to convert these logits into a probability distribution across the classes. The loss is then computed by comparing the predicted class probabilities with the ground truth labels, which are provided as integer class indices for each pixel. In cases where the dataset exhibits class imbalance, the loss function can be weighted accordingly to give more importance to underrepresented classes, thereby reducing the likelihood of the model ignoring minority classes during training.

In all cases, a dropout of 0.2 is implemented at the bottleneck of the network (i.e., 20% of the neurons are "switched off" to limit overfitting).

5.3 Input Fields Processing

The following describes the pre-processing that was applied to the images representing the inputs that will be used by the network, namely the measurements made by WIVERN of the Doppler velocity field and radar reflectivity. This pre-processing was found to be useful during the various initial iterations of the study of the problem.

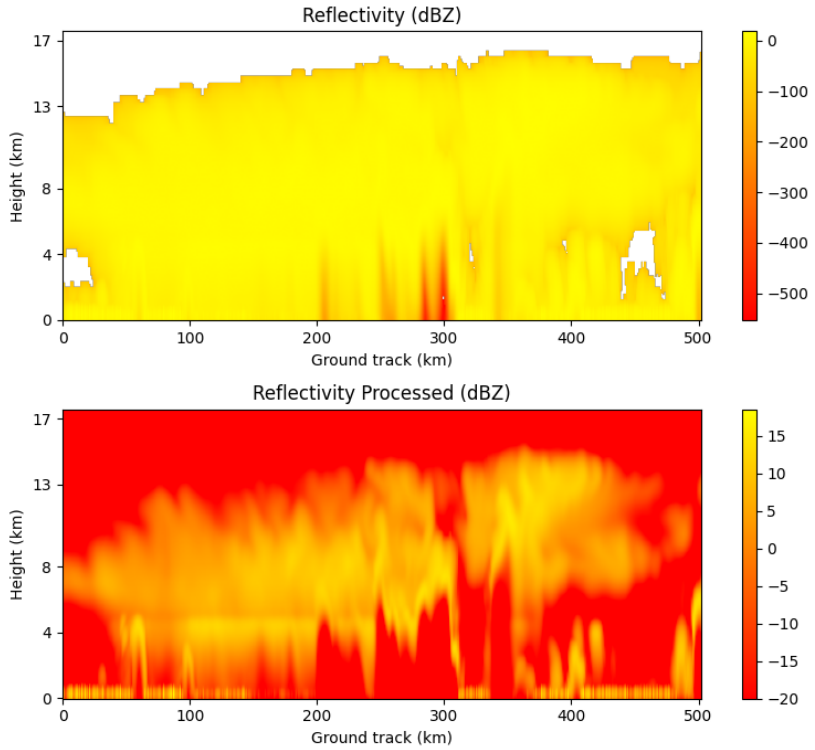


Figure 10: Reflectivity field. On top there is the raw file, on the bottom there is the processed image before the normalization.

- For the **reflectivity** field, NaN values were replaced with -20 dBZ. Values lower than approximately -20 dBZ are associated with areas that do not belong to cloud structures and are therefore not of interest. Consequently, all pixels with values lower than this threshold were also set to -20 dBZ. The reflectivity field was then normalised to the [0, 1] interval in order to secure a better stability during the training process.

- For the **Doppler velocity**, similarly, NaN values were replaced with the neutral value of 0 m/s. A mask based on radar reflectivity was then applied, setting to 0 m/s all pixels where the reflectivity was less than or equal to -20 dBZ. This mask serves to clean the Doppler velocity field of measurements taken outside the cloud structures, which consist of noise. Finally, the absolute value of the Doppler velocity field was normalised to the [0, 1] interval in order to improve generalization of the information in the field.

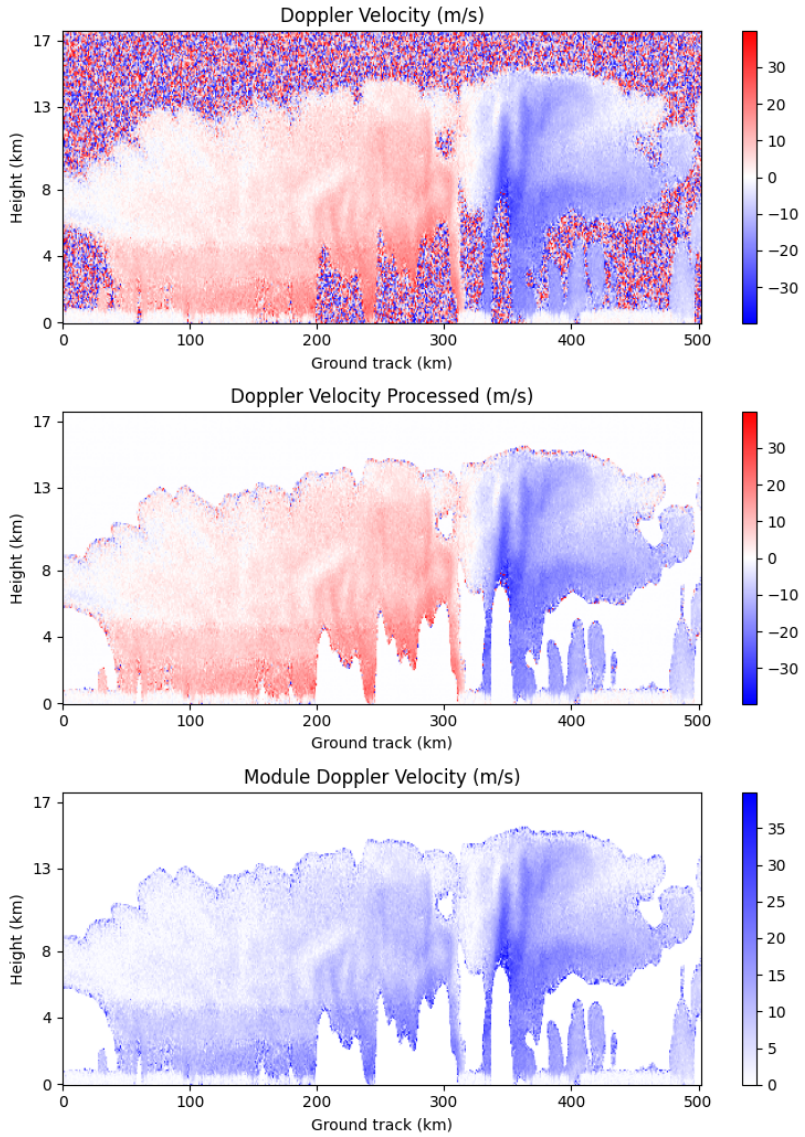


Figure 11: Doppler velocity field. On top there is the raw file, on the bottom there is the processed image before the normalization.

5.4 Output Target Processing

One of the most important aspects in developing a reliable model is performing appropriate processing of the target data that the model will see during training. These target images are, in fact, what the model should learn to replicate once it is queried after training. All target images are based on the vertical velocity field along the line of sight, available from simulations, and on the reflectivity field. Several case studies have been addressed, and for each of them, the target images may vary depending both on the type of classification or segmentation desired, and on how different processing strategies can improve the model's training.

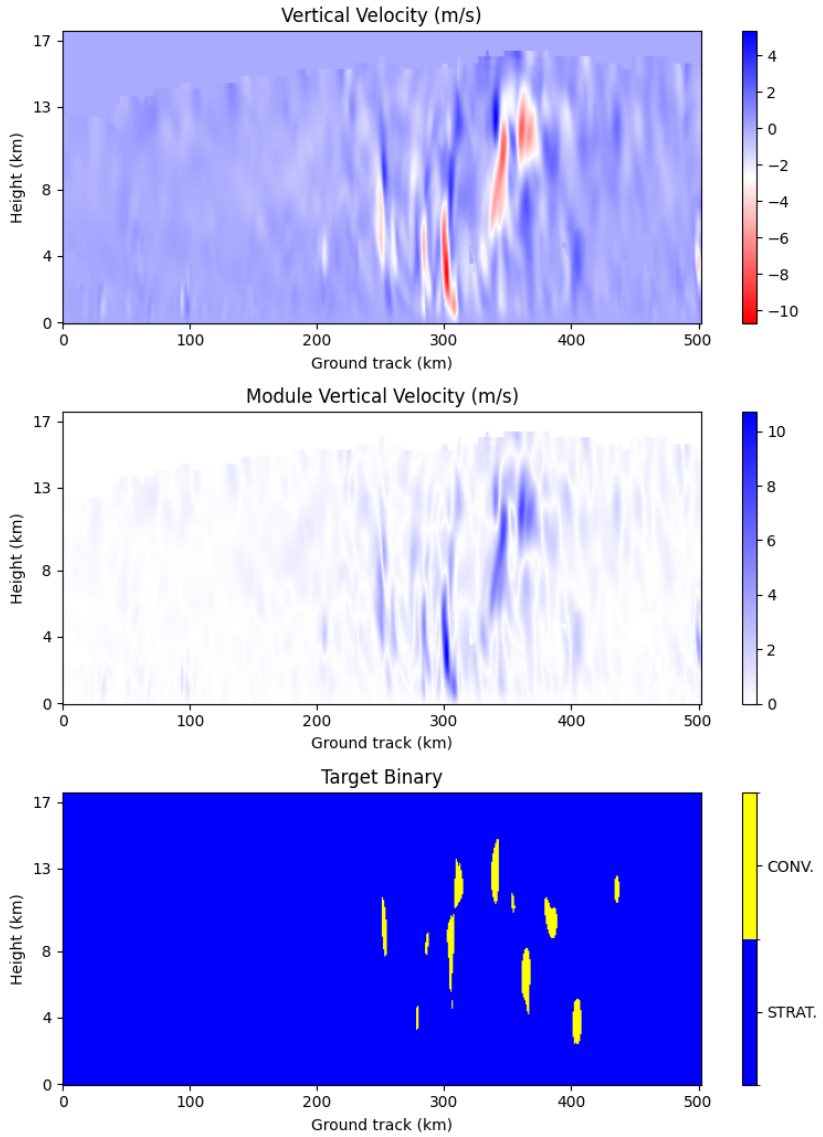


Figure 12: Steps for the generation of the target map from the vertical velocity field for binary segmentation. In this case the target is discrete. The yellow pixels are the convective cases.

- **Binary segmentation**. For this case study, the aim is to train a model capable of recognising convective areas within the observed scene and producing as output a map that directly highlights them. The target images, which the network will need to learn to reproduce, are binary maps in which each pixel (t) is assigned a value according to the corresponding class. In this case, we are dealing with binary segmentation, as there are two classes to distinguish: the background or stratiform areas and the convective areas. The construction of these target images is carried out by applying a

threshold to the magnitude of the vertical velocity field w (Figure 12).

$$t(w,\, z) = \begin{cases} 1 & \text{if } z > -20 \text{ dBZ} \;\; \text{and } w > 1.5 \text{ m/s} \\ 0 & \text{if } z <= -20 \text{ dBZ} \;\; \text{or } w <= 1.5 \text{ m/s} \end{cases}$$

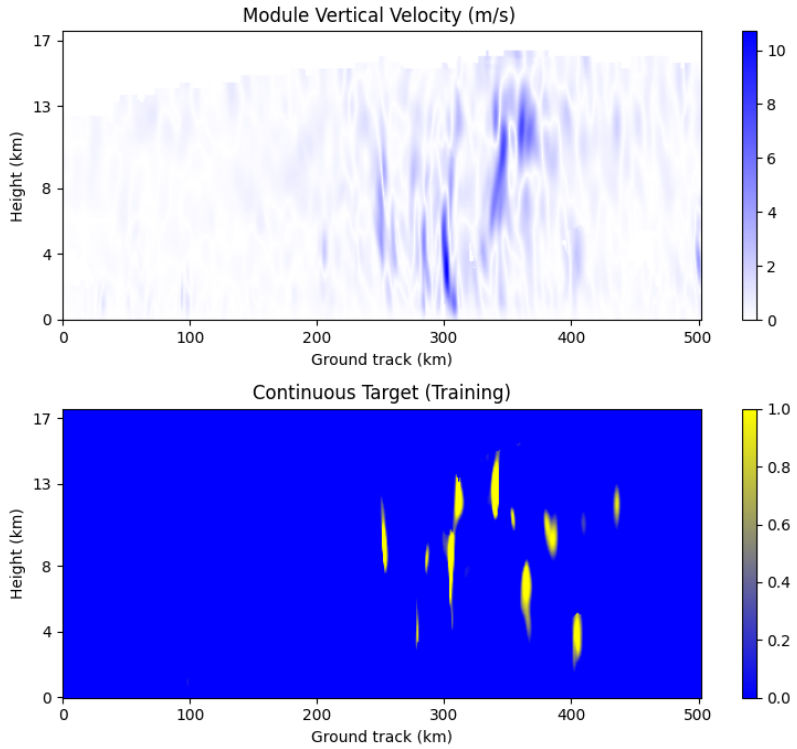


Figure 13: The processing of the vertical velocity in order to obtain the continuous mask for the training and the binary target for the evaluation and metric calculation.

- **Binary segmentation with continuous mask**. The aim to observe how the model can improve if the target is not represented as a binary map with only discrete values, but rather as a probability field. In the previous case, the target maps contained only 0 and 1 to represent the two possible cases that the network must learn to recognise and classify for each pixel. In this case, however, the target map also contains values between 0 and 1 to represent the transition region between the two classes in terms of probability. A binary map with discrete values can later be obtained by applying a threshold to the continuous target constructed for this case study. Below are the definitions used for constructing the probability maps with reference to vertical velocity (w) and reflectivity (z). A linear function has been chosen to represent the probability transition between stratiform and convective within the specified velocity interval. Other functions may be

employed, as well as wider or narrower interval, depending on how strictly one wishes to define the convective case (Figure 13).

$$t(w,\, z) = \begin{cases} 0 & \Longrightarrow |w| < 1 \text{ m/s} \;\; \text{or} \;\; z > -20 \text{ dBZ} \\ |w| - 1 & \Longrightarrow 1 \text{ m/s} \leq |w| \leq 2 \text{ m/s} \;\; \text{and} \;\; z > -20 \text{ dBZ} \\ 1 & \Longrightarrow |w| > 2 \text{ m/s} \;\; \text{and} \;\; z > -20 \text{ dBZ} \end{cases}$$

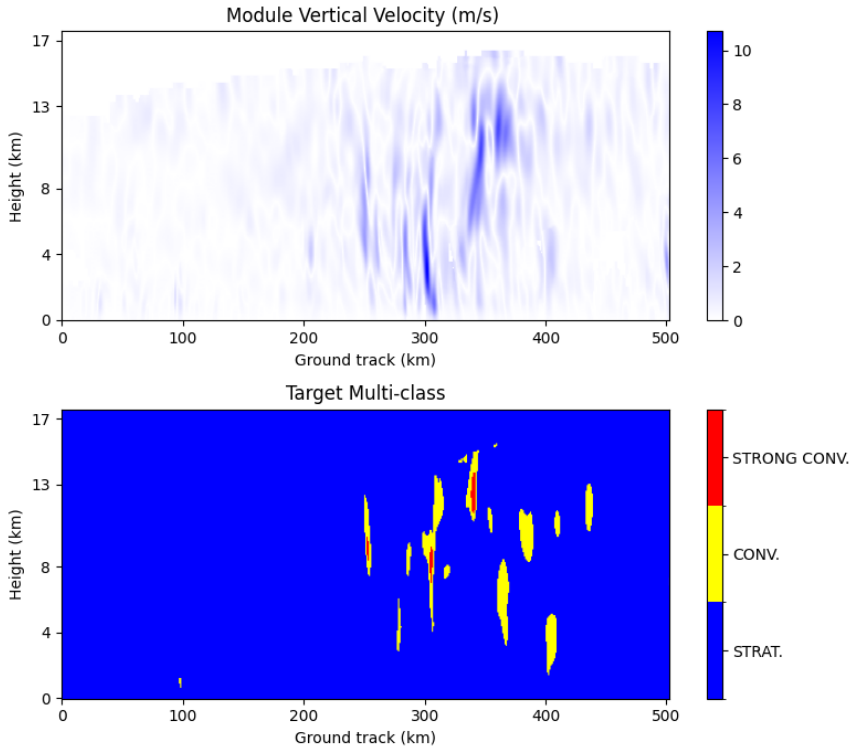


Figure 14: Example of multi-class target image for training ad the masked image used foe the evaluation.

- **Multi-class segmentation**. The following case study involves training a model capable of performing multi-class classification. In this scenario, the network is not only required to distinguish between convective and stratiform classes, but also to differentiate between multiple types of convection, as defined specifically for this case study.

  In particular, three classes are defined in the construction of the target images (Figure 14):

  - **Class 0** (Stratiform): defined according to the reflectivity and vertical velocity conditions described below. It can also be considered as the background class and is by far the most frequent class within the dataset.

$$|w| < 1\,\text{m/s} \;\; \textit{and} \;\; z > -20 \text{ dBZ} \tag{5}$$

- **Class 1** (Weak Convection): represents regions where convection begins to appear and can be interpreted as a transitional area between stratiform regions and stronger convective zones. Its frequency in the dataset is significantly lower than that of Class 0.

$$1\,\mathrm{m/s} \leq |w| < 2.5\,\mathrm{m/s} \quad \textit{and} \quad z > -20\ \mathrm{dBZ} \tag{6}$$

- **Class 2** (Strong Convection): associated with the most intense convective events and the highest vertical velocities. Consequently, it is the least frequent class in the dataset—more so than in any of the previous cases addressed.

$$|w| \geq 2.5\,\mathrm{m/s} \quad \textit{and} \quad z > -20\ \mathrm{dBZ} \tag{7}$$

6 Evaluation

The trained models are evaluated on the images belonging to the validation dataset, which are therefore not used for training the model. The predictions made by the network from the input fields are compared with the targets constructed according to the specific case study. A qualitative evaluation is carried out by observing the differences between the two resulting images — that is, the predicted image and the target image. A quantitative evaluation is performed through the calculation of metrics in order to obtain an overall view of the performance of the trained model. The main metrics calculated are listed below and are based on the values of the confusion matrix.

- **POD** (**Probability of Detection**), also called **Recall**, expresses the fraction of positive pixels that the model manages to identify. It is not a reliable estimate as it does not take false positives into account. Low values indicate that positive pixels are not being recognised by the model.

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

- **FAR** (**False Alarm Rate**) expresses the fraction of false positives relative to the number of active pixels in the prediction made by the model. High values indicate that the network assigns many pixels to the object class that do not actually belong to it.

$$FAR = \frac{FP}{FP + TP}$$

- **BIAS** compares the number of active pixels in the prediction with those in the target. Values close to 1 indicate that the network is close to reproducing the number of active pixels present in the target, although it is not guaranteed that these are in the same position. Values much greater than 1 indicate that the model tends to over-predict the object within the image, while the opposite is true for very low values close to zero.

$$Bias = \frac{TP + FP}{TP + FN}$$

- **ETS** (Equitable Threat Score) takes into account the "hits" obtained by chance and is therefore a more reliable metric. It penalises both false positives and false negatives.

$$ETS = \frac{TP - TP_{ref}}{TP + FP + FN - TP_{ref}} \quad \text{where } TP_{ref} = \frac{(TP + FN) \times (TP + FP)}{TP + FN + FP + TN}$$

- **F1 Score** (**F1**) is the harmonic mean of precision and recall, and provides an overall assessment of the model's quality by considering both the avoidance of false positives and false negatives. A high F1 score means that the U-Net has both good precision and good recall.

$$F1 = \frac{2 \times TP}{2 \times TP + FP + FN}$$

- **Precision** expresses the proportion of positive pixels that are actually positive. High values indicate that the model tends not to make mistakes when classifying a pixel as positive.

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

The evaluation metrics are calculated on the validation dataset, which consists of images not used during training. Before computing these metrics, however, both the target and predicted images are appropriately processed. Specifically, the metrics are not calculated over all pixels of the image, but only over those that meet a reflectivity threshold condition.

$$z > 20 \text{ dBZ} \tag{8}$$

As a result, not all pixels in the image are considered valid; a mask is therefore applied to exclude from the calculation any pixels that do not satisfy this condition (8). An example of this processing, applied to both the predictions and the targets of the dataset, is shown in Figure 15. Once the valid pixels have been identified for each image, a confusion matrix is constructed for each pair of target and prediction, calculating the values of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). The values reported for each metric correspond to the average values obtained across all image pairs in the validation dataset.

6.1 Binary Segmentation Results

Below are the metrics used to evaluate the trained models. As previously mentioned, the results of the various iterations carried out during the training phase are reported, in order to obtain the most accurate model and to observe its behaviour as certain parameters vary. In general, one of the most important aspects influencing the network's performance is the dataset used for training. An appropriate representation of the features and of the class to be recognised is crucial for both the training process and the model's performance. Since class 0 is much more frequent in the dataset compared to the under-represented convective class (Class 1), the model will learn very quickly and easily to classify pixels as belonging to class 0. For this reason, one of the first aspects we aimed to investigate

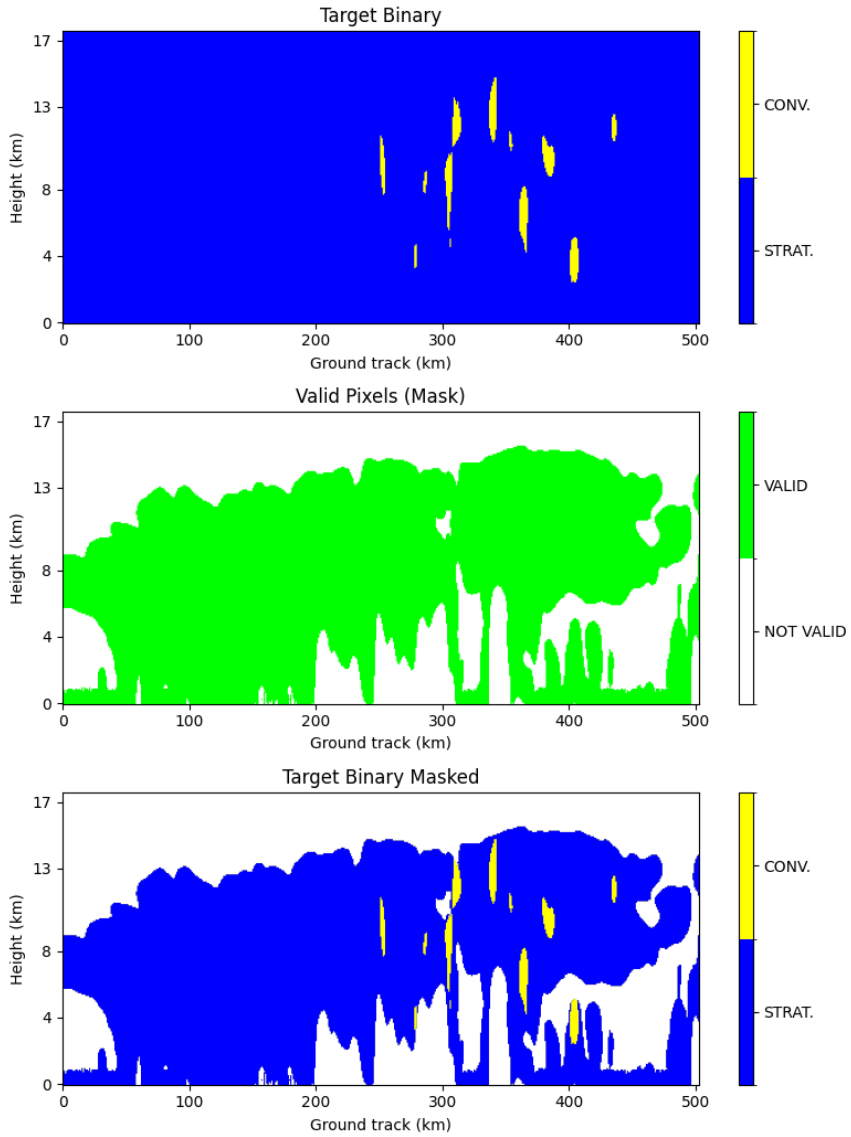


Figure 15: Processing for evaluation. It is represented the application of the mask based on the reflectivity in order to have the valid pixels for the metrics calculation. The green pixels are those whose reflectivity meets condition 8.

was the effect of the weight assigned to the positive class in the loss function. Starting from $w = 1$, which implies a neutral impact of the weight on the loss function during training, its value was gradually increased. Higher values of w penalise false negatives more heavily and force the network to attempt to predict the under-represented positive class in the dataset. Of course, excessively high values tend to produce models that over predict the positive class. The results are presented in the following tables (Table 4 and 5).

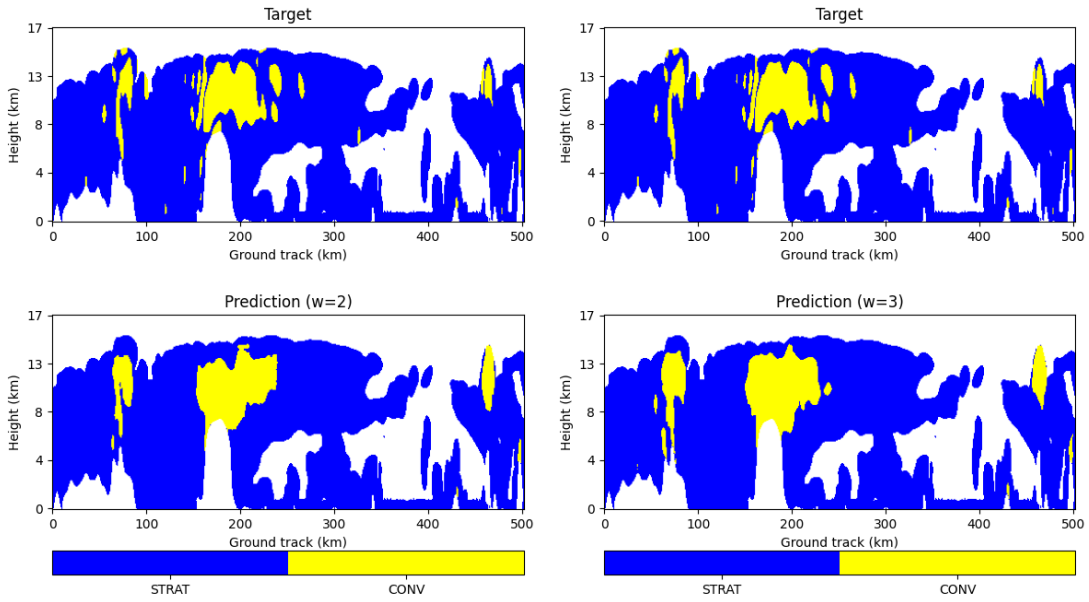


Figure 16: Best validation loss models. On the left it is used $w = 2$, on the right $w = 3$.

.

w	POD	FAR	Bias	ETS	F1	Precision	Recall
2	0.5454	0.4401	1.1106	0.3570	0.4971	0.5599	0.5454
3	0.6092	0.4935	1.4026	0.3578	0.5005	0.5065	0.6092

Table 4: The metrics refer to the models trained with different weights (w) for the loss function with the best validation loss.

w	POD	FAR	Bias	ETS	F1	Precision	Recall
2	0.5278	0.4614	1.1003	0.3446	0.4846	0.5386	0.5278
3	0.5952	0.5109	1.4090	0.3484	0.4909	0.4891	0.5952

Table 5: The metrics refer to the models trained with different weights (w) for the loss function. These are obtained after all the epochs of the training.

From the metrics calculated in the highlighted cases (Table 4), it is interesting to note that varying the weight value does not necessarily lead to an improvement across all metrics. With $w = 2$, the model tends to under-predict convective events associated with the pixels, as indicated by a lower POD. However, this comes with the advantage of a lower FAR, suggesting that the model is also less prone to false positives and behaves more cautiously in classification. This is also reflected in the Bias, which is closer to 1. On the other hand, with $w = 3$, we observe higher

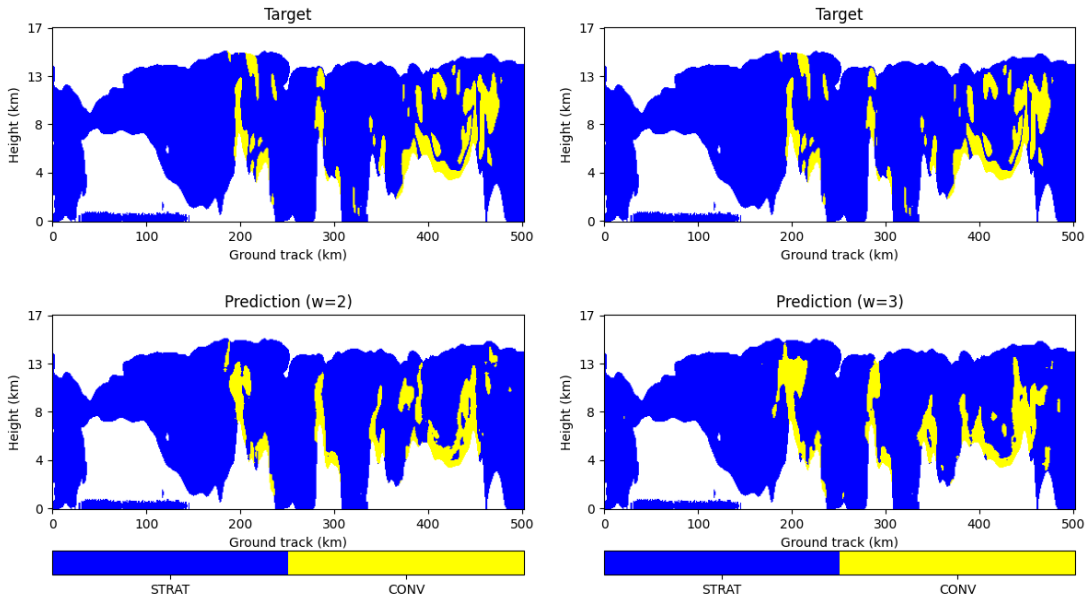


Figure 17: Models results after the whole training cycle. On the left it is used $w = 2$, on the right $w = 3$.

.

values. Precision tends to be better with a lower weight, albeit at the expense of a poorer recall. A model with a higher weight tends to improve recall and thus detects a larger portion of the positive target pixels, whereas precision is better with a lower weight, meaning the model tends to minimise false positives.

To gain an overall understanding of the optimal value of w to use, other metrics such as ETS and F1 score are considered. The ETS value is generally acceptable, indicating that the model predictions are better than random guesses and that features in the images are beginning to be recognised. As the weight increases, its value improves slightly, as does the F1 score. The harmonic mean between precision and recall gives an idea of the most advantageous trade-off between the two metrics, and even if only slightly, with $w = 3$ the results are better.

Comparing the results obtained (Table 4) with those associated with the models trained for the entire training cycle (Table 5), we note that all metrics (except for the Bias, which is best for $w = 2$ compared to all other iterations) have worsened. According to these values, the models obtained at the end of the full training cycle have degraded in performance due to overfitting. From a qualitative perspective, it is noticeable from the images that the structures of larger convective cells are detected, and microstructures are often highlighted as well. Despite the worsening metrics, it is noteworthy that in Figure 17, all the main convective phenomena have been recognised and segmented in a very similar shape.

A larger model (introduced in the previous chapter) was trained and its metrics

were then compared with those of the smaller model, with all training parameters kept the same. The larger model includes one additional layer, and the initial layer of the encoder contains twice as many feature maps (increased from 32 to 64). The number of parameters to be computed becomes considerably higher and, as a result, so does the computational cost. In fact, the training time more than tripled, increasing from 2 hours and 45 minutes to approximately 10 hours.

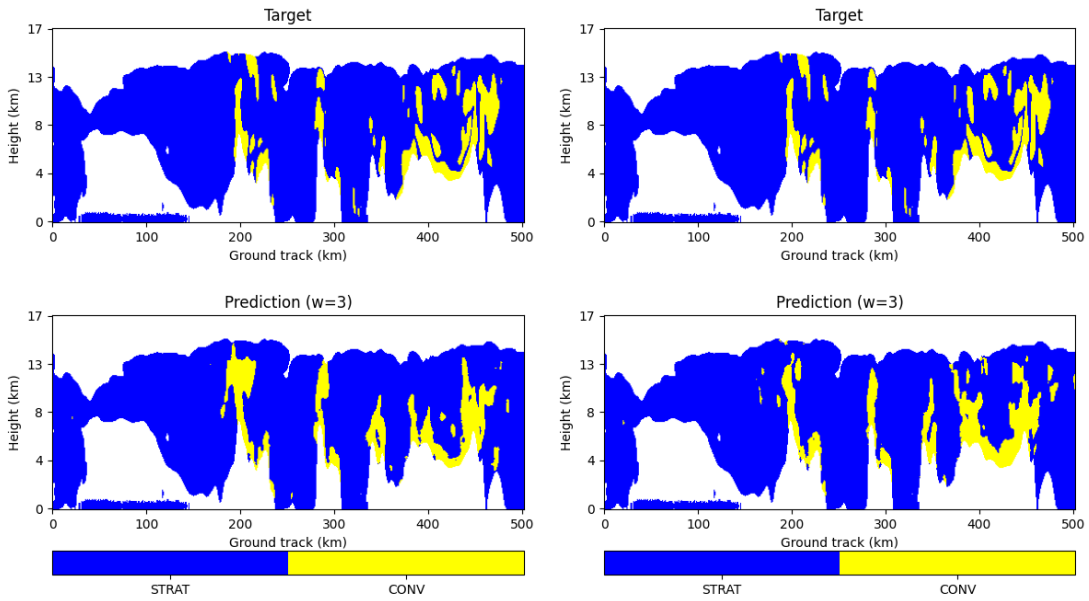


Figure 18: On the left the result of the smaller model, on the right of the larger one.

.

Size	POD	FAR	Bias	ETS	F1	Precision	Recall
Small	0.6092	0.4935	1.4026	0.3578	0.5005	0.5065	0.6092
Large	0.6193	0.4912	1.3861	0.3645	0.5083	0.5088	0.6193

Table 6: The metrics refer to two different size models, both with $w = 3$.

As expected, all metrics improve when using a larger model. Not only does the number of events recognised by the network increase, but false positives also decrease. The network's predictions begin to be more accurate, as both precision and recall increase, and consequently, the F1 score improves as well. The ETS inevitably rises, indicating a model with overall better performance. However, it should be noted that the metric improvements are not significant, and from a qualitative perspective, they are not easily appreciable. As shown in Figure 10, the detection of convective events remains reliable, and in some cases, the segmentation of certain contours improves. An increase in model size and number

of parameters certainly ensures improved recognition of more complex patterns and greater accuracy, but it is also important to consider the computational cost required for training, and whether this cost is justified for this particular task.

An evaluation was carried out to assess how much the two inputs could affect the performance of the trained model. In this regard, keeping the same selected weight value, two networks were trained using only one of the two available observables at a time.

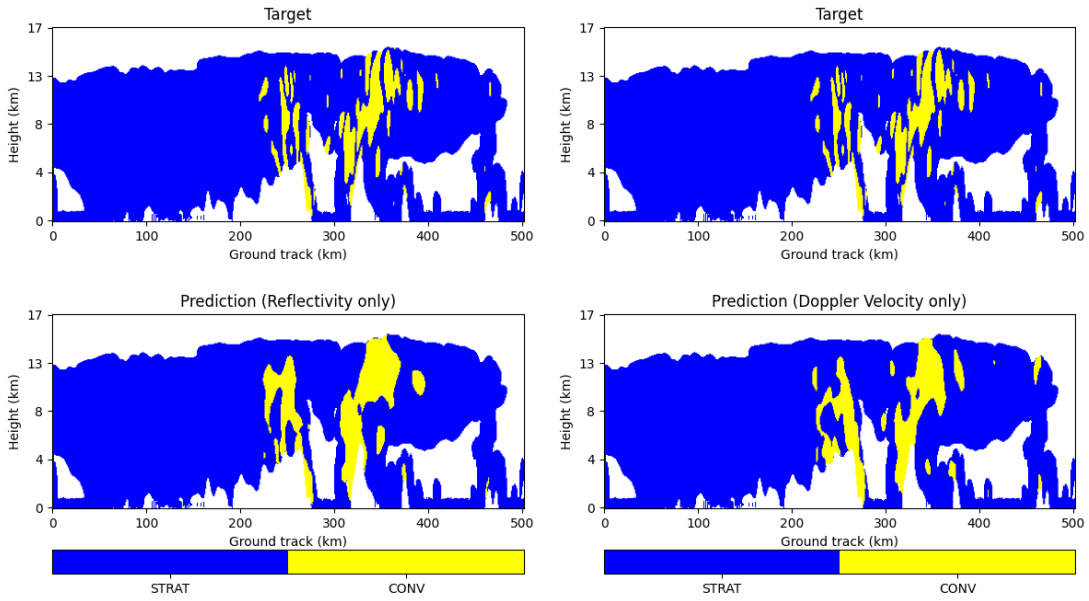


Figure 19: Results using only one input field. On the left it has used only the reflectivity field, on the right only the Doppler velocity field

.

Input	POD	FAR	Bias	ETS	F1	Precision	Recall
VDP	0.4385	0.5374	1.1753	0.2659	0.3951	0.4626	0.4385
DBZ	0.5160	0.4373	1.0173	0.3506	0.4888	0.5627	0.5160

Table 7: The metrics refers to models trained with only one type of observable as input. VDP is the the model trained with only the Doppler velocity, while DBZ is the one trained with only the reflectivity.

From the metric results obtained (Table 7), it is evident that the variable with the greatest influence on model training is reflectivity. Doppler velocity alone does not yield acceptable results, as it not only results in a lower POD but also a higher FAR. Although the bias is close to 1 in both cases—indicating that events are neither underpredicted nor overpredicted—the model trained solely

with reflectivity shows higher precision and recall. This leads to an F1 score that is approximately 10% higher, though still lower than that of models trained with both inputs. The ETS, which provides a more general assessment of model performance, also has a higher value for the model trained with reflectivity alone. From a qualitative perspective, by observing the images (Figure 11), we note that smaller-scale events are not identified individually; instead, the model tends to merge the convective structures present in the scene.

6.2 Binary Segmentation with Continuous Target Results

In this case study, for the evaluation phase, post-processing of the images is carried out, since the validation metrics are based on the comparison of binarised images. The continuous targets in the validation dataset are transformed into discrete binary targets by applying a threshold value to distinguish between the two classes. What changes is the reduced distance between the network predictions and the targets (during the training process), which are now continuous rather than discrete. For the network predictions, the sigmoid function is applied to the model's output, thereby producing a continuous heatmap with values between 0 and 1, which is then converted into a discrete binary map by applying a threshold value of 0.5.

Model	POD	FAR	Bias	ETS	F1	Precision	Recall
Discrete	0.5719	0.4616	1.2082	0.3607	0.5016	0.5384	0.5719
Continuous	0.6154	0.4854	1.4087	0.3641	0.5070	0.5146	0.6154

Table 8: The metrics refers to the small model trained with continuous values target map.

The table 8 reports a comparison between the metrics computed for the model trained with binary discrete targets and the one trained with continuous targets. From the values presented, it can be observed that the model trained with continuous targets shows a better ability to detect actual events (higher sensitivity) in this case study. Both models generate a similar number of false alarms and tend to overestimate events, particularly the model with continuous targets. The F1 score is slightly higher for the model with discrete targets, indicating a better balance between recall and precision. The ETS is also slightly higher for the continuous-target model, suggesting a generally well-balanced overall performance.

In conclusion, for this case study, the model trained with continuous targets tends to detect a greater number of events but also produces more false alarms. From a qualitative perspective, the images (figure 20) show that the model demonstrates a better ability to segment even smaller and isolated events, as well as to approximate the larger convective structures present in the scene.

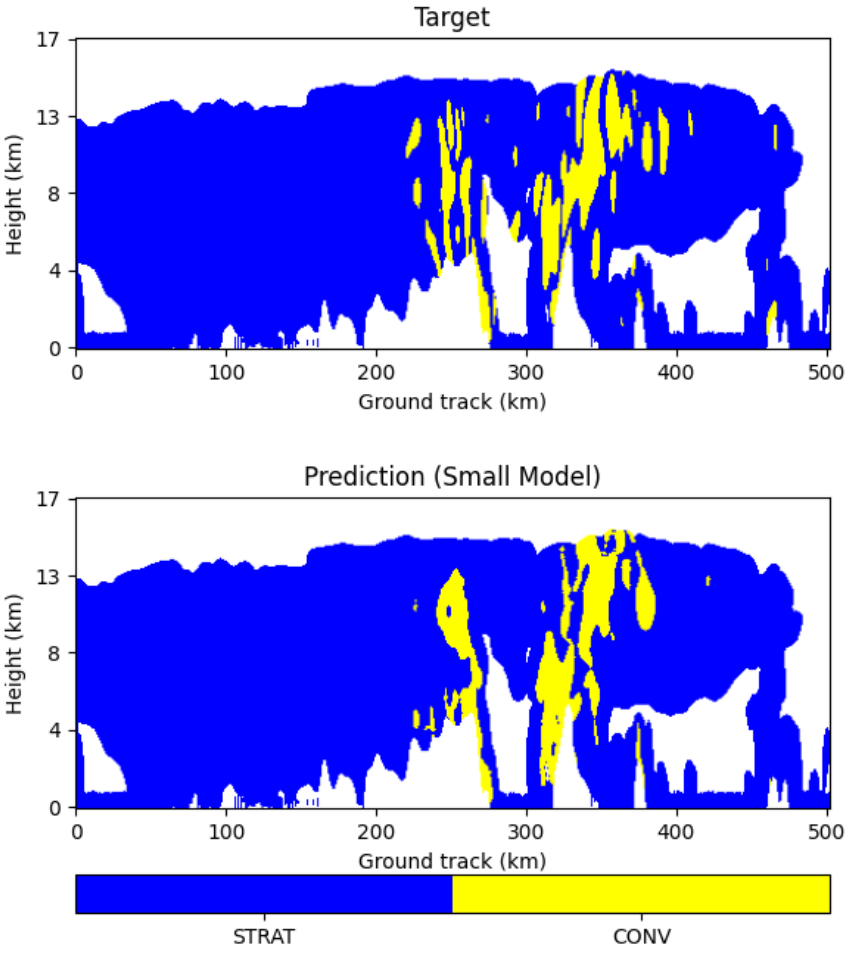


Figure 20: The prediction is compared with the ground truth for the model trained using a continuous mask.

6.3 Multi-class Segmentation Results

The objective of this case study is to evaluate the performance of a model trained to recognize more than two classes. In particular, in addition to the stratiform class, there are also the convective and strong convective classes. For this case study, it was important to analyse how the model's performance varies according to the weight values assigned in the loss function.

It is important to note that, in the binary segmentation setup, the positive class (Class 1) was already under-represented. Now, for this multi-class case study, that class is further divided into two subclasses, which are therefore even more under-represented in the dataset. As a result, it becomes essential to understand which weights should be assigned to the loss function for each class the model must learn to detect and segment.

From the early stages of training, Class 2, to which a weight of $w_2 = 8$ was assigned, showed good performance metrics. Consequently, the focus shifted to identifying the optimal value of w for Class 1. Multiple training sessions of the same model were therefore conducted, varying the weight w_1.

For each of these training runs, average metric values were calculated for both classes.

Model	POD	FAR	Bias	ETS	F1	Precision	Recall
$w_1 = 5 \vert w_2 = 8$	0.5283	0.6428	1.7682	0.3405	0.3863	0.3572	0.5283
$w_1 = 3 \vert w_2 = 8$	0.4663	0.5959	1.2849	0.3501	0.3934	0.4041	0.4663
$w_1 = 2 \vert w_2 = 8$	0.4039	0.5493	0.9877	0.3434	0.3828	0.4507	0.4039
$w_1 = 1 \vert w_2 = 8$	0.2769	0.4666	0.5461	0.3063	0.3288	0.5334	0.2769

Table 9: Performance Metrics for Class 1.

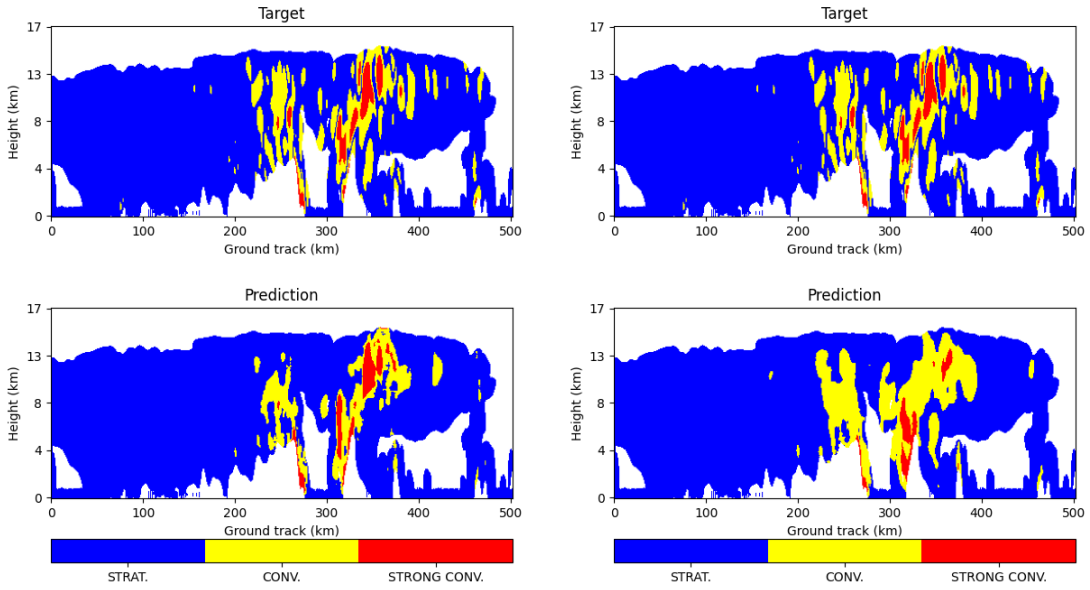
Model	POD	FAR	Bias	ETS	F1	Precision	Recall
$w_1 = 5 \vert w_2 = 8$	0.5702	0.4252	1.1456	0.3799	0.5058	0.5748	0.5702
$w_1 = 3 \vert w_2 = 8$	0.6183	0.4679	1.3422	0.3792	0.5055	0.5321	0.6183
$w_1 = 2 \vert w_2 = 8$	0.6507	0.5045	1.5106	0.3704	0.4986	0.4955	0.6507
$w_1 = 1 \vert w_2 = 8$	0.6914	0.5422	1.7750	0.3615	0.4895	0.4578	0.6914

Table 10: Performance Metrics for Class 2.

For Class 1 (Table 9), it is observed that as the weight assigned to this class increases, the POD (Probability of Detection) tends to rise. This means that the model is able to detect more positive events belonging to this class. However, this increase is accompanied by a rise in the FAR (False Alarm Rate), indicating a greater number of false alarms, i.e., incorrect positive predictions – and not false negatives, as previously mistakenly indicated. The model with a weight of $w_1 = 2$ for Class 1 shows a bias very close to 1, suggesting near-optimal calibration: the model neither overestimates nor underestimates the frequency of Class 1 events. From this perspective, it is the most balanced among those analysed. The ETS (Equitable Threat Score) and F1 Score metrics for Class 1 are generally low across all models, suggesting that predicting this class is inherently more difficult. Among all configurations, the model with weight $w_1 = 3$ represents a good compromise, showing the highest values for both metrics, at the cost of a slight loss in calibration.

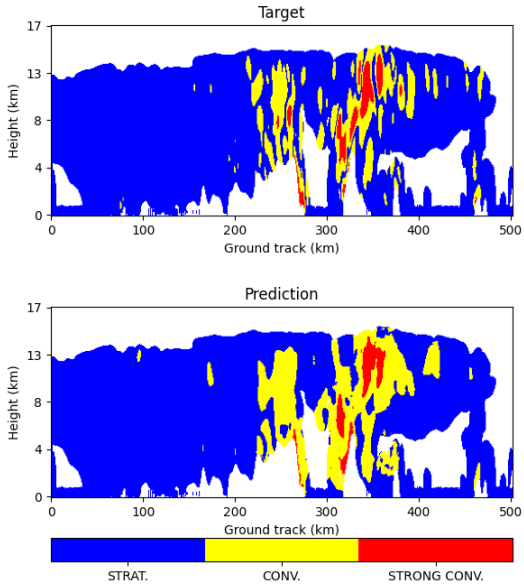
Regarding Class 2 (Table 10), the metrics are significantly better than those for Class 1. The $[w_1 = 5 \vert w_2 = 8]$ model, which corresponds to a higher weight assigned to Class 1 ($w_1 = 5$), achieves the highest ETS and F1 Score values for Class 2, indicating excellent overall performance in this category.

An interesting aspect is how changing the weight for Class 1 indirectly affects the performance of Class 2. The model with the best calibration for Class 2

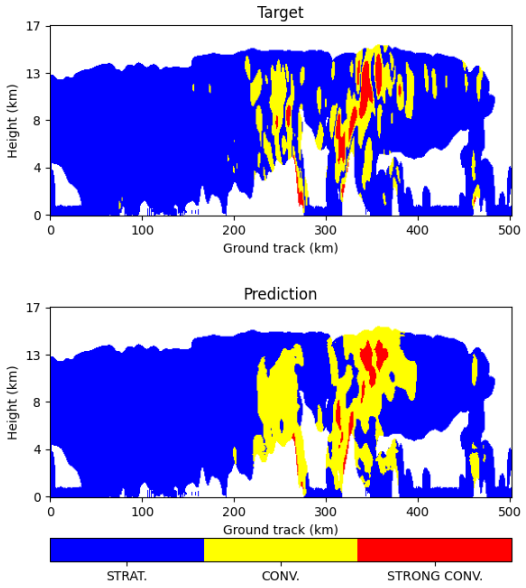


(a) $w_1 = 1$, $w_2 = 8$

(b) $w_1 = 2$, $w_2 = 8$



(c) $w_1 = 3$, $w_2 = 8$



(d) $w_1 = 5$, $w_2 = 8$

Figure 21: Predictions of all the iterations for the multi-class segmentation.

(bias closest to 1), namely $[w_1 = 5|w_2 = 8]$, tends to overestimate Class 1, as indicated by its high bias for that class. This reflects a tension between the two classes: improving performance for one may penalize the other. The aggregate metrics for Class 2, such as ETS and F1 Score, show only slight variations as the weight for Class 1 changes. However, the best model for Class 2 ($[w_1 =$

$5|w_2 = 8]$) turns out to be suboptimal for Class 1. Therefore, when considering the overall balance between the two classes – especially given the challenges in predicting Class 1 – an intermediate weight such as $w_1 = 3$ appears to be the most effective compromise, offering good overall performance and a balanced trade-off. This analysis highlights the importance of jointly evaluating performance across different classes, especially in the context of imbalanced classification. The definition of the "best" model strongly depends on the application goals and the importance assigned to different types of errors.

7 Conclusion

This work aimed to develop an Artificial Intelligence model capable of distinguishing, within simulated radar images from the WIVERN mission, convective regions from stratiform ones. The underlying idea is simple: to harness the power of deep learning to automate a complex task that typically requires meteorological expertise or the use of threshold-based algorithms or tabulated values.

To tackle this problem, a well-known architecture in the field of image segmentation was chosen: the U-Net. This network proved particularly suitable for the radar meteorology context thanks to its ability to capture both the overall structure of the images and their finer details. In fact, the results obtained with binary segmentation—i.e., simply distinguishing convective from non-convective areas—showed that the model was quite effective at correctly identifying many relevant structures.

One of the main strengths of the work was precisely the use of the U-Net architecture: its characteristic "U"-shaped structure, with direct connections between encoding and decoding layers, allowed for acceptable predictions even in the presence of complex or isolated events. Originally developed for biomedical applications, this type of network proved to be just as versatile in this context.

Another key element was the generation of training data using an end-to-end simulator, which allowed the model to be trained on realistic meteorological scenarios (such as the case study of Hurricane Milton) without needing to wait for real satellite observations. This approach turned out to be not only practical but also very effective in providing a wide range of examples—something essential for training a neural network properly.

The tests carried out showed that the most informative observable is radar reflectivity, which alone was able to guide the model effectively. Doppler velocity contributed less when used on its own, but when combined with reflectivity, it enhanced model performance in a synergistic way.

Among the most effective strategies used during the project was the implementation of a weighted loss function, which helped address the strong class imbalance in the dataset. Convective regions, in fact, make up only a tiny fraction of the data compared to stratiform ones. Without this adjustment, the model would likely have ignored the convective class almost entirely. Thanks to class weighting, the network was pushed to detect even weak or isolated signs of convection.

The use of continuous masks instead of discrete binary labels also helped the network learn the transition from stratiform to convective more gradually, improving overall prediction quality, particularly in ambiguous situations.

Finally, the project also explored a multi-class approach, distinguishing between weak convection, strong convection, and stratiform areas. While more

complex and challenging to train due to the underrepresentation of certain classes, this method produced promising results that deserve further exploration.

That said, a few limitations emerged. The most significant was the severe class imbalance: convective areas are so scarce relative to the total dataset that even well-trained models tended to underestimate them. To address this, future work could employ targeted data augmentation techniques to artificially increase the frequency of convective regions, or use more balanced sampling strategies.

Another difficulty encountered was overfitting in the more complex models. Adding depth and parameters led to only marginal improvements in performance, but significantly increased training time and reduced the model's generalization on validation data. In this case, it would be helpful to apply techniques such as early stopping, more aggressive dropout, or regularization to make the model more robust.

Lastly, while larger models were promising, they also introduced high computational costs. With an eye toward possible operational use—such as real-time application or on-board satellite processing—it would be worth considering lighter and more efficient architectures.

In conclusion, this work laid the groundwork for applying Artificial Intelligence to the classification of convective structures in spaceborne radar data. The developed model demonstrated acceptable segmentation capabilities and adapted well to complex scenarios. Although there is still room for improvement, the strategies adopted—both in terms of architecture and data preparation—pointed in the right direction.

This approach has strong potential for future integration with real data from the WIVERN mission, contributing not only to better segmentation and classification models, but also to improved numerical weather prediction and a deeper understanding of atmospheric processes on a global scale.

References

[1] Zhang. "The vision of space-based component of WMO Integrated Global Observing Systems (WIGOS) in 2040 Anticipating requirements and new space technologies." In: *EUMETSAT Meteorological Satellite Conf* (2016).

[2] Nicolas Sasso et al. "Impact of WIVERN Wind Observations on ARPEGE Numerical Weather Prediction Model Forecasts Using an Ensemble of Data Assimilation Method". In: *Quarterly Journal of the Royal Meteorological Society* n/a.n/a (), e4991. DOI: https://doi.org/10.1002/qj.4991.

[3] A. J. Illingworth et al. "WIVERN: A New Satellite Concept to Provide Global In-Cloud Winds, Precipitation, and Cloud Properties". In: *Bulletin of the American Meteorological Society* 99.8 (2018), pp. 1669 –1687. DOI: 10.1175/BAMS-D-16-0047.1. URL: https://journals.ametsoc.org/view/journals/bams/99/8/bams-d-16-0047.1.xml.

[4] ESA WIVERN-RfA. *WIVERN Report for Assessment*. Tech. rep. available at https://eo4society.esa.int/event/earth-explorer-11-user-consultation-meeting/. ESA-EOPSM-WIVE-RP-4375, 2023.

[5] A. Battaglia et al. "Observation error analysis for the WInd VElocity Radar Nephoscope W-band Doppler conically scanning spaceborne radar via end-to-end simulations". In: *Atmospheric Measurement Techniques* 15.9 (2022), pp. 3011–3030. DOI: 10.5194/amt-15-3011-2022. URL: https://amt.copernicus.org/articles/15/3011/2022/.

[6] M. Wolde et al. "Implementation of polarization diversity pulse-pair technique using airborne W-band radar". In: *Atmospheric Measurement Techniques* 12.1 (2019), pp. 253–269. DOI: 10.5194/amt-12-253-2019. URL: https://amt.copernicus.org/articles/12/253/2019/.

[7] Olivier Pauluis, Arnaud Czaja, and Robert Korty. "The Global Atmospheric Circulation on Moist Isentropes". In: *Science* 321.5892 (2008), pp. 1075–1078. DOI: 10.1126/science.1159649. URL: https://www.science.org/doi/abs/10.1126/science.1159649.

[8] T. N. Krishnamurti et al. "A Pathway Connecting the Monsoonal Heating to the Rapid Arctic Ice Melt". In: *Journal of the Atmospheric Sciences* 72.1 (2015), pp. 5 –34. DOI: 10.1175/JAS-D-14-0004.1. URL: https://journals.ametsoc.org/view/journals/atsc/72/1/jas-d-14-0004.1.xml.

[9] Robert A. Houze Jr. et al. "The variable nature of convection in the tropics and subtropics: A legacy of 16years of the Tropical Rainfall Measuring Mission satellite". In: *Reviews of Geophysics* 53.3 (2015), pp. 994–1021. DOI: https://doi.org/10.1002/2015RG000488. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2015RG000488.

[10] Kamil Mroz et al. “Hail-Detection Algorithm for the GPM Core Observatory Satellite Sensors”. In: *Journal of Applied Meteorology and Climatology* 56.7 (2017), pp. 1939 –1957. DOI: 10.1175/JAMC-D-16-0368.1. URL: https://journals.ametsoc.org/view/journals/apme/56/7/jamc-d-16-0368.1.xml.

[11] John M. Haynes et al. “Rainfall retrieval over the ocean with spaceborne W-band radar”. In: *Journal of Geophysical Research: Atmospheres* 114.D8 (2009). DOI: https://doi.org/10.1029/2008JD009973. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2008JD009973.

[12] Alessandro Battaglia et al. “Multiple-scattering in radar systems: A review”. In: *Journal of Quantitative Spectroscopy and Radiative Transfer* 111.6 (2010), pp. 917–947. ISSN: 0022-4073. DOI: https://doi.org/10.1016/j.jqsrt.2009.11.024. URL: https://www.sciencedirect.com/science/article/pii/S0022407309003677.

[13] Matthias Steiner, Robert A. Houze, and Sandra E. Yuter. “Climatological Characterization of Three-Dimensional Storm Structure from Operational Radar and Rain Gauge Data”. In: *Journal of Applied Meteorology and Climatology* 34.9 (1995), pp. 1978 –2007. DOI: 10.1175/1520-0450(1995)034<1978:CCOTDS>2.0.CO;2. URL: https://journals.ametsoc.org/view/journals/apme/34/9/1520-0450_1995_034_1978_ccotds_2_0_co_2.xml.

[14] Yi Yang, Xin Chen, and Youcun Qi. “Classification of convective/stratiform echoes in radar reflectivity observations using a fuzzy logic algorithm”. In: *Journal of Geophysical Research: Atmospheres* 118.4 (2013), pp. 1896–1905. DOI: https://doi.org/10.1002/jgrd.50214. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/jgrd.50214.

[15] John L. Cintineo et al. “A Deep-Learning Model for Automated Detection of Intense Midlatitude Convection Using Geostationary Satellite Images”. In: *Weather and Forecasting* 35.6 (2020), pp. 2567 –2588. DOI: 10.1175/WAF-D-20-0028.1. URL: https://journals.ametsoc.org/view/journals/wefo/35/6/waf-d-20-0028.1.xml.

[16] Leo Thomas Ramos and Angel D. Sappa. “Leveraging U-Net and selective feature extraction for land cover classification using remote sensing imagery”. In: *Scientific Reports* 15.1 (2025), p. 784. ISSN: 2045-2322. DOI: 10.1038/s41598-024-84795-1. URL: https://doi.org/10.1038/s41598-024-84795-1.

[17] Chuan Yan et al. “Improved U-Net Remote Sensing Classification Algorithm Based on Multi-Feature Fusion Perception”. In: *Remote Sensing* 14.5 (2022). ISSN: 2072-4292. DOI: 10.3390/rs14051118. URL: https://www.mdpi.com/2072-4292/14/5/1118.

[18] Lei Han et al. “Convective Precipitation Nowcasting Using U-Net Model”. In: *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022), pp. 1–8. DOI: 10.1109/TGRS.2021.3100847.

[19] Jannik Hoeller et al. “U-Net Segmentation for the Detection of Convective Cold Pools From Cloud and Rainfall Fields”. In: *Journal of Geophysical Research: Atmospheres* 129.1 (2024). e2023JD040126 2023JD040126, e2023JD040126. DOI: https://doi.org/10.1029/2023JD040126. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2023JD040126.

[20] J. Meza, N. Anderson, and T. Tsultrim. “Forecasting UNET: Leveraging a U-Net Architecture for Probabilistic Tornado Prediction from Multi-variate Reanalysis Data”. In: *Weather* 60 (2024), pp. 1–8.

[21] C. M. Nguyen et al. “Coincident in situ and triple-frequency radar airborne observations in the Arctic”. In: *Atmospheric Measurement Techniques* 15.3 (2022), pp. 775–795. DOI: 10.5194/amt-15-775-2022. URL: https://amt.copernicus.org/articles/15/775/2022/.

[22] F. E. Scarsi et al. “Mispointing characterization and Doppler velocity correction for the conically scanning WIVERN Doppler radar”. In: *Atmospheric Measurement Techniques* 17.2 (2024), pp. 499–514. DOI: 10.5194/amt-17-499-2024. URL: https://amt.copernicus.org/articles/17/499/2024/.

[23] Alessandro Battaglia et al. “I and Qs Simulation and Processing Envisaged for Spaceborne Polarization Diversity Doppler Radars”. In: *IEEE Transactions on Geoscience and Remote Sensing* 63 (2025), pp. 1–14. DOI: 10.1109/TGRS.2025.3529672.

[24] Alessandro Battaglia et al. “Characterization of Surface Radar Cross Sections at W-Band at Moderate Incidence Angles”. In: *IEEE Transactions on Geoscience and Remote Sensing* 55.7 (2017), pp. 3846–3859. DOI: 10.1109/TGRS.2017.2682423.

[25] Graeme Stephens et al. “The First 30 Years of GEWEX”. In: *Bulletin of the American Meteorological Society* 104.1 (2023), E126 –E157. DOI: 10.1175/BAMS-D-22-0061.1. URL: https://journals.ametsoc.org/view/journals/bams/104/1/BAMS-D-22-0061.1.xml.

[26] Hanii Takahashi, Zhengzhao Johnny Luo, and Graeme L. Stephens. “Level of neutral buoyancy, deep convective outflow, and convective core: New perspectives based on 5years of CloudSat data”. In: *Journal of Geophysical Research: Atmospheres* 122.5 (2017), pp. 2958–2969. DOI: https://doi.org/10.1002/2016JD025969. URL: https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1002/2016JD025969.