universität wien

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Nowcasting of wind profiles along the approach corridors of Vienna International Airport based on machine learning

verfasst von | submitted by

Yousif Isam George Rasho BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien | Vienna, 2025

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 066 614
Studienrichtung lt. Studienblatt | Degree programme as it appears on the student record sheet:	Masterstudium Meteorology and Climate Science
Betreut von | Supervisor:	Ass.-Prof. Mag. Dr. Manfred Dorninger

Abstract

The nowcasting of wind profiles is a crucial aspect of aviation meteorology, particularly for ensuring safe and efficient landings. This thesis focuses on the development and evaluation of a machine learning-based model to predict wind profiles along the approach corridors of Vienna International Airport (VIE). The research leverages Mode S aircraft-derived wind data, which provides high-resolution atmospheric information. Deep learning architecture, like long short-term memory (LSTM) networks, was employed to analyze and forecast wind speed and direction at different altitudes.

The study begins by addressing the challenges of data acquisition and preprocessing, emphasizing the importance of accurate wind profile extraction. The dataset, compiled from historical aircraft telemetry and meteorological sources, is carefully filtered and structured to ensure high model reliability. Key statistical metrics, such as mean absolute error (MAE) and root mean squared error (RMSE), are used to evaluate the model's performance.

Results show that the model performs well under stable meteorological conditions, with predictions closely matching observed wind patterns. However, significant deviations occur during rapidly changing weather phenomena, such as frontal passages and turbulence events. Analysis of error metrics reveals that wind speed predictions exhibit lower discrepancies compared to wind direction forecasts, which are more susceptible to sudden shifts in atmospheric conditions.

The findings highlight the potential of AI-driven nowcasting models to improve situational awareness in air traffic management. The integration of real-time aircraft data with advanced machine learning techniques offers new possibilities for enhancing predictive capabilities in meteorological forecasting, ultimately contributing to safer and more efficient aviation operations.

Zusammenfassung

Das Nowcasting von Windprofilen ist ein entscheidender Aspekt der Luftfahrtmeteorologie, insbesondere zur Gewährleistung sicherer und effizienter Landungen. Diese Arbeit konzentriert sich auf die Entwicklung und Bewertung eines auf maschinellem Lernen basierenden Modells zur Vorhersage von Windprofilen entlang der Anflugkorridore des Flughafens Wien (VIE). Die Forschung nutzt Winddaten, die aus Mode-S-Flugzeugdaten abgeleitet wurden und hochauflösende atmosphärische Informationen liefern. Dabei kommt eine Deep-Learning-Architektur wie Long Short-Term Memory (LSTM)-Netzwerke zum Einsatz, um Windgeschwindigkeit und -richtung in verschiedenen Höhen zu analysieren und vorherzusagen.

Die Studie beginnt mit der Darstellung der Herausforderungen bei der Datenerfassung und -vorverarbeitung und betont die Bedeutung einer genauen Extraktion von Windprofilen. Der Datensatz, der aus historischen Flugzeugtelemetriedaten und meteorologischen Quellen zusammengestellt wurde, wird sorgfältig gefiltert und strukturiert, um eine hohe Zuverlässigkeit des Modells sicherzustellen. Wichtige statistische Kennzahlen wie der mittlere absolute Fehler (MAE) und der quadratische Mittelwertfehler (RMSE) werden verwendet, um die Leistungsfähigkeit des Modells zu bewerten.

Die Ergebnisse zeigen, dass das Modell unter stabilen meteorologischen Bedingungen gute Leistungen erbringt, wobei die Vorhersagen den beobachteten Windmustern nahekommen. Bei rasch wechselnden Wetterphänomenen wie Frontdurchgängen und Turbulenzereignissen treten jedoch deutliche Abweichungen auf. Die Analyse der Fehlerkennzahlen zeigt, dass die Vorhersagen der Windgeschwindigkeit geringere Abweichungen aufweisen als jene der Windrichtung, die empfindlicher auf plötzliche Veränderungen der atmosphärischen Bedingungen reagiert.

Die Ergebnisse unterstreichen das Potenzial KI-gestützter Nowcasting-Modelle zur Verbesserung des Situationsbewusstseins im Flugverkehrsmanagement. Die Integration von Echtzeit-Flugdaten mit fortschrittlichen Methoden des maschinellen Lernens eröffnet neue Möglichkeiten zur Verbesserung der Prognosefähigkeiten in der meteorologischen Vorhersage und trägt letztlich zu sichereren und effizienteren Luftfahrtoperationen bei.

Acronyms & Abbreviations

ACF Autocorrelation Function

ACG Austrocontrol GmbH

AD Aerodrome

ADB-S Automatic Dependent Surveillance - Broadcast

AI Artificial Intelligence

AIP Aeronautical Information Publication

AIRAC Aeronautical Information Regulation And Control

ATM Air Traffic Management

CET Central European Time

CEST Central European Summer Time

CNN Convolutional Neural Network

DNN Deep Neural Network

G/P Glide path

GRU Gated Recurrent Unit

gs Ground Speed

IAS Indicated Airspeed

kts Knots

LOWW Vienna International Airport (ICAO)

LSTM Long Short-Term Memory

MAE Mean Absolute Error

Mode-S Mode-select

MSE Mean Squared Error

netCDF Network Common Data Form

RNN Recurrent Neural Network

RMSE Root Mean Square Error

RWY Runway

SSR Secondary Surveillance Radar

tas True Airspeed

VIE Vienna International Airport (IATA)

Contents

1. Introduction

Wind has several effects on aviation; for example, it affects aircraft performance during take-off, landing, and cruise. Headwinds can increase the time and distance required for take-off and landing, while tailwinds can reduce them. Crosswinds can affect the aircraft's stability during these critical phases. Wind shear refers to sudden changes in wind speed and direction over a short distance. Wind shear poses a significant hazard during take-off and landing, as it can affect aircraft performance and control (Golding, 2005).
The aim of this thesis is to build a model, which can make a nowcasting of wind based on artificial intelligence to predict the wind profiles for landing aircraft along the approach corridor of Vienna International Airport (VIE). The idea of this work was developed through Austrocontrol GmbH (ACG), which provided the data and the resources.

1.1 Motivation

Wind profiles refer to patterns and characteristics of wind speed and direction at a specific location or over a given period of time. Wind profiles can vary according to factors such as geographical location, local topography, and atmospheric conditions. The intensification of Clear Air Turbulence (CAT) and wind shears can also have an impact on aircraft safety (Klimenko and Krozel, 2009). Hence, wind profiles are important to forecast especially for approach corridors, so that Air Traffic Management (ATM) can specify which of these approach corridors they should take in operation and therefore which runway would be used (Liu et al., 2020). The idea of predicting the wind based on artificial intelligence is to improve and test the abilities of the machine to assure whether it can help to reduce the efforts that humans can have during the operations.

1.2 Problem statement

Making a time series forecast of wind profiles based on machine learning algorithms could be difficult for the following reasons:

- Wind vectors are derived and calculated from a Mode S transponder on each aircraft that is equipped with such an instrument (Sun et al., 2017). In this case if an aircraft does not have a Mode S transponder, there will be no wind data available. The availability of the wind data may vary in time and space, e.g. A breakdown in the radar system for a specific time frame can lead to no data coverage or for example, during the night hours the air traffic is reduced to nearly zero, this lead also to reduced or even missing data coverage. Therefore, there will be missing values in the dataset, which need to be filled with an imputation method (see section 2.2) or ignored.
- Signals that are sent from the Secondary Surveillance Radar (SSR) to the Mode S transponder can sometimes be jammed or shadowed, for example, due to the different signal paths of each approach corridor. It also makes the preprocessing challenging for sorting and filtering the data, because Vienna International Airport (VIE) has different approach corridors and depending on the aircraft's position in the air, some fixed directions or approach corridors relative to SSR have always disruptions (e.g. arrivals on Runway 29). This also limits data availability.

1.3 Extraction of wind vector

The ATC SSR on the ground usually transmits a signal to the aircraft, which contains a request for some data parameters, e.g. ICAO ID, altitude, Mach number, true airspeed (tas), ground speed (gs), heading, etc. The aircraft transmits all this information to an advanced Mode-S receiver on the SSR or to a local ADB-S / Mode-S receiver on the airfield or other locations (Sun et al., 2018).
The horizontal wind vector ($\overrightarrow{V_w}$) as shown in figure 1, can be obtained from the two calculated velocities ($\overrightarrow{V_{tas}}$ & $\overrightarrow{V_{gs}}$)*:

$$\overrightarrow{V_w} = \overrightarrow{V_{gs}} - \overrightarrow{V_{tas}} \tag{1}$$

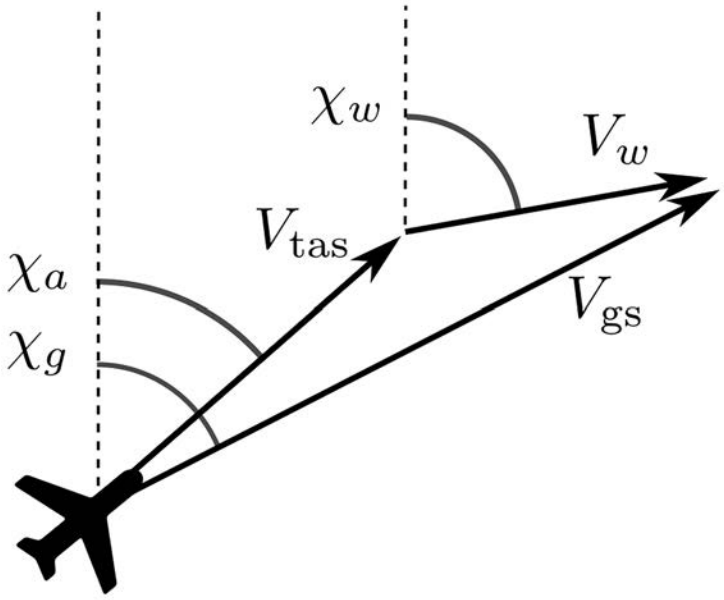


Figure 1: Angles χ_a, χ_g and χ_w with respect to the true north refer to aircraft heading, true angle and wind direction respectively (Sun et al., 2018).

1.4 Application of the wind vector

The extracted wind vector $\overrightarrow{V_w}$ can be calculated along the flight path of an aircraft for each specific distance depending on the SSR transmission. In this case, all calculated $\overrightarrow{V_w}$ will result in a wind profile, which depends on time and height.
The $\overrightarrow{V_w}$ can be calculated at each height and location, but in this thesis, only the necessary part of the flight path (the final approach path) will be considered in the model, which is crucial for the ATM and therefore for the forecast of the wind profiles.

*for the calculation of $\overrightarrow{V_{tas}}$ & $\overrightarrow{V_{gs}}$ refer to (Sun et al., 2018)

1.5 Section overview

In section 2, the data will be described in detail including the origin of the data and the data preprocessing steps until having a complete prepared dataset for the application as input data in the model. Section 3 gives an introduction and overview about artificial intelligence (AI), the algorithms and the choice of the most useful and meaningful model for the thesis and lastly the verification methods to estimate the accuracy of the predictions against the observations. The results will be presented in section 4, where the outputs of the model represent the nowcasting of wind speed and direction. The results are shown as wind profile plots for selected interesting weather conditions and statistically in the verifications. Finally, section 5 gives a discussion and review of the results and an outlook on what can be improved or developed in the future.

2. Data

In the world of modern aviation, data acquisition and analysis play a pivotal role in ensuring safety, efficiency, and precision. In this context, Mode S data stands out as a vital source of information, providing detailed information on aircraft movements, trajectories, and atmospheric conditions. In this chapter, we dive into the significance of Mode S data, particularly focusing on its provision by ACG. Our primary focus revolves around one specific data parameter: wind profiles. Wind profiles, extracted from Mode S data, serve as invaluable input for predictive modeling, especially when integrated with AI algorithms. This chapter sets the stage for understanding the extraction of wind profiles, their provider, and the utilization of wind profiles for AI-driven wind prediction models. Through this exploration, the aim is to underscore the critical role of data in enhancing aviation operations and safety standards.

2.1 Wind profiles

The wind profiles that are extracted from Mode S data will be converted and saved as netCDF files. Each netCDF file represents a one-day dataset, which has multiple key parameters, e.g. time (t), altitude (h), RWY, pressure (p), wind speed (ff), wind direction (dd), temperature (T), etc.
Each key parameter depends on specific variables, e.g. the wind speed depends on the time, RWY, and the altitude. Data are sorted as daily files consisting of 96 time steps, each time step represents a 15-min interval (i.e. 24 h). The RWY parameter consists of 4 variables, which represent the 2 runways that are used in both directions. The altitude parameter consists of 64 height steps up to nearly 3150 m (~10000 ft). Therefore, each time step (15 min) has 64 height steps. The 15-minute time step includes all aircraft that have flown in this time step, where the maximum number of aircraft used the specific RWY in this 15-minute time interval, and the key parameters are averaged over the time step. In this case, the dataset can be constructed as a data frame, which has the shape of an array of 96 x 64. One time and one height step result in one pixel (value). Since the aim of this study is to forecast the wind profiles, only the wind speed (ff) and the wind direction (dd) will be used. Figure 2 represents the examples of these data frames for 1 July 2019.

2.2 Data preprocessing

The original data, as shown, need to be filtered and adjusted before using it as input into the neural networks. Some of the data have to be excluded due to their unavailability or unreliability, which might affect the forecasting process later. For example, the first few height steps from the ground will be excluded due to radar signal noise near RWY 29, especially in the time around 5/6 UTC (7 CET/CEST) and after 19/20 UTC (21 CET/CEST), when RWY 29 is often used due to northwesterlies and the Noise Abatement Procedure*. The route of the approaching aircraft may vary from each other depending on ATC decisions, but generally, there are approach routes containing multiple waypoints, which are defined by the civil aviation authority of the country and are listed in the AIP as a standard approach procedure. Pilots must fly those routes unless otherwise instructed by ATC. Those procedures are called STAR (Standard Terminal Arrival)†.

*refer to the AIP of Austria for more routes details: `https://eaip.austrocontrol.at` (02.05.2024)
†`https://www.faa.gov/air_traffic/publications/atpubs/aim_html/chap5_section_4.html` (02.05.2024)

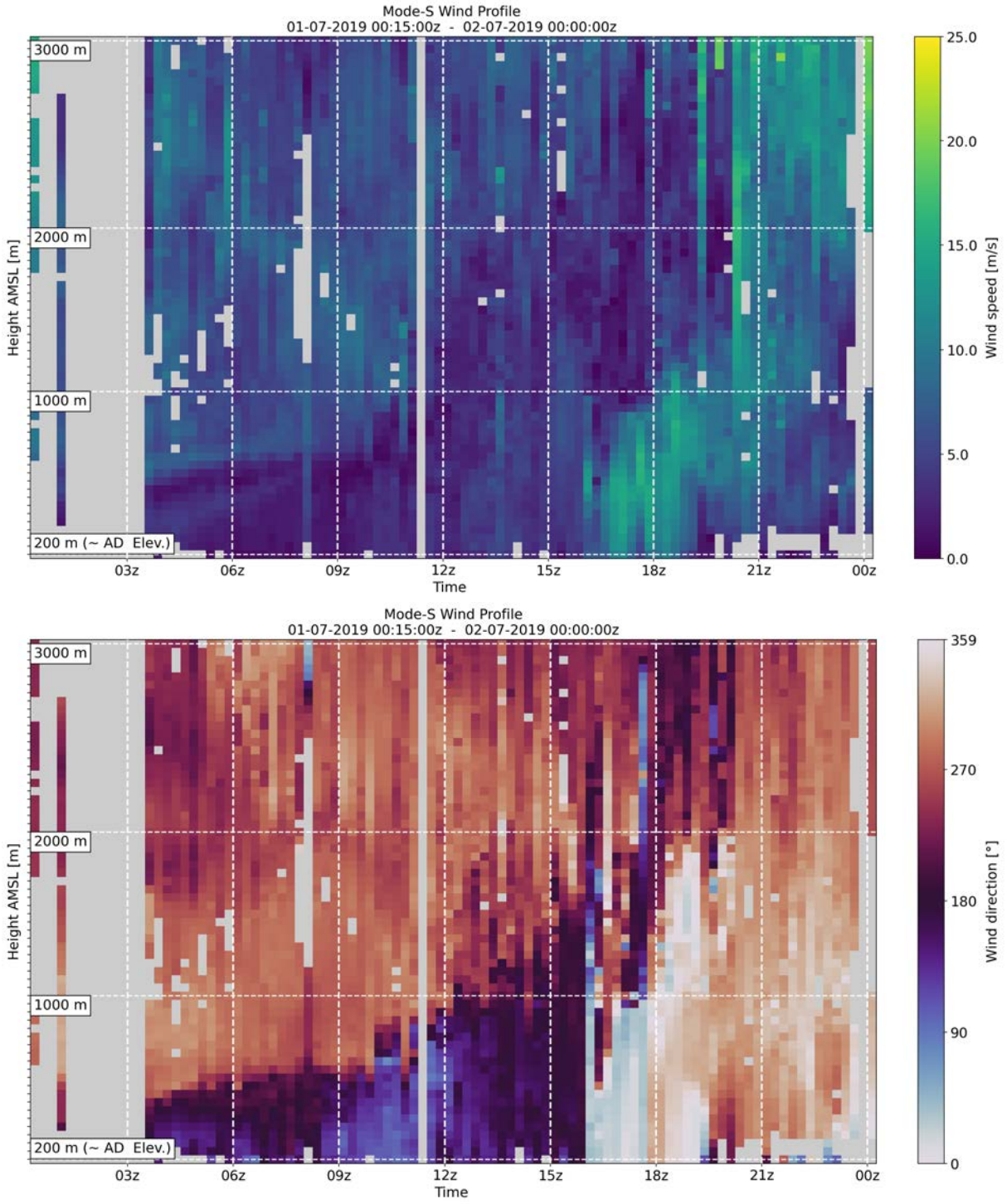


Figure 2: Wind speed (ff) given in [m/s] for 1 July 2019 (upper). The time on the x-axis represents 24 hours in UTC. The height AMSL on the y-axis are given in meters. The grey shaded areas represent the missing data. Wind direction (dd) given in degrees [°] (lower).

After flying a STAR procedure, another procedure can follow, which is called a "Transition". The transition is only a connecting route between STAR and IAP (Instrument Approach Procedure). Each RWY has a different transition. Usually, the ATC gives the pilots shortcuts, which lead them to the IAP [‡]. The IAP is the final approach phase before landing, where the aircraft becomes closer and closer to the surface until landing on the RWY. The IAP includes the final glide path, where the aircraft should only descend in a straight line with a standard glide slope of 3° (it might vary depending on the location of the aerodrome (AD) and its surrounding terrains)(Ribbens, 2003).
The IAP is usually a mandatory route for each aircraft approaching the AD to use, no matter which type it is. The beginning of an IAP is always an FAP (Final Approach Point), and this FAP could have different heights, depending on the RWY glide path and the terrain below (Federal Aviation Administration (FAA), 2017). According to the AIP charts of Austria, LOWW has FAP heights of 3000 ft or 5000 ft (914.4 m or 1524 m), depending on which RWY is in use.
Due to the multiple STARs and transitions, the aircraft can approach the AD from different directions, which can lead to different wind values, e.g. in the case of a cold front or inversion. Hence, the upper wind data above 1500 m mainly flown as STAR procedure and/or transitions are excluded from the analysis. Figure 3 shows the new structure of the data frame.
During the night hours (from 22 to 5 UTC), the air traffic is usually reduced, or at some hours, there are no landing aircraft at all. Hence, there are no data available in this period. The data unavailability, in general, will be represented as NAN values (Not a Number) in the netCDF data, and with those NAN values, the model cannot be trained precisely. Therefore, data between 22 and 5 UTC will be excluded from the dataset (see fig. 4).

Finally, the last part of the preprocessing chapter describes the process of filling the remaining missing data (Imputation).
In a first step, wind speed (ff) and wind direction (dd) need firstly to be converted into u & v components with the following equations (Grange, 2014):

$$u = -|\vec{V}|sin\,\phi \tag{2}$$

$$v = -|\vec{V}|cos\,\phi \tag{3}$$

Now the dataset gets a range of positive and negative values, which is advantageous for the application of the imputation (see fig. 5).
Values above or below ± 40 m/s are considered as outliers and set to NAN values. In this case, they are nothing but missing values, which will be filled later by the imputation method.
The imputation method is only applied for missing data in the time period between 5 and 22 UTC (grey pixels in Fig 4) and is based on using the median filter, which involves filling in missing values in a dataset based on the median values of their neighbors. In order to use the median filter, the NAN values need to be replaced with a value; this value is the mean of the dataset per day. The function $'scipy.ndimage.median-filter'$ in Python applies a median filter to an input array, which can be used to impute missing values by replacing each missing value with the median of its neighboring values (Arnaud et al., 2023).
The function has multiple arguments; the first two are important to define which are the dataset input and the size. The size of the window used for median filtering should be chosen based on the data distribution and expected noise characteristics. A window that is too small has not effectively filled in missing values, while, on the other hand, a window that is too large has smoothed out important

[‡] https://eaip.austrocontrol.at/lo/250418/Charts/LOWW/LO_AD_2_LOWW_13-1-3_en.pdf (05.05.2025)

data features[§]. Multiple attempts at choosing the size of the window have been made, where finally, window size 4 showed the best result. After executing the code, one gets a fully filled dataset as shown in figure 6.
Generally, the data in each AI model are usually split into training, validation, and test sets. The reason for splitting the data is to ensure that the model is robust and performs well. The data in this study are split as follows:

1. **Training set:** this set is used for training the model and consists of 70% of the entire dataset.
2. **Validation set:** this set is used for fine-tuning the hyper parameters and avoids overfitting. It consists of 15% of the entire dataset.
3. **Test set:** it is used for the model's performance evaluation. It makes 15% of the rest of the entire dataset.

After splitting and saving all data for both the u- and v-components, the training, validation, and test datasets need to be normalized to ensure that all features have a similar scale. This can help the model to train faster and improve accuracy. It also reduces computational problems caused by very large or small values.
The numerical computation of the normalization $\hat{x}$ is very simple and is represented in equation 4. It is done by subtracting the mean μ of the training dataset values for each height step from each of the three original split dataset values x divided by the standard deviation σ of the training dataset values for each height step.

$$\hat{x} = \frac{x - \mu}{\sigma} \tag{4}$$

[§]refer to SciPy API for code details `https://docs.scipy.org/doc/scipy/reference/generated/scipy.ndimage.median_filter.html`

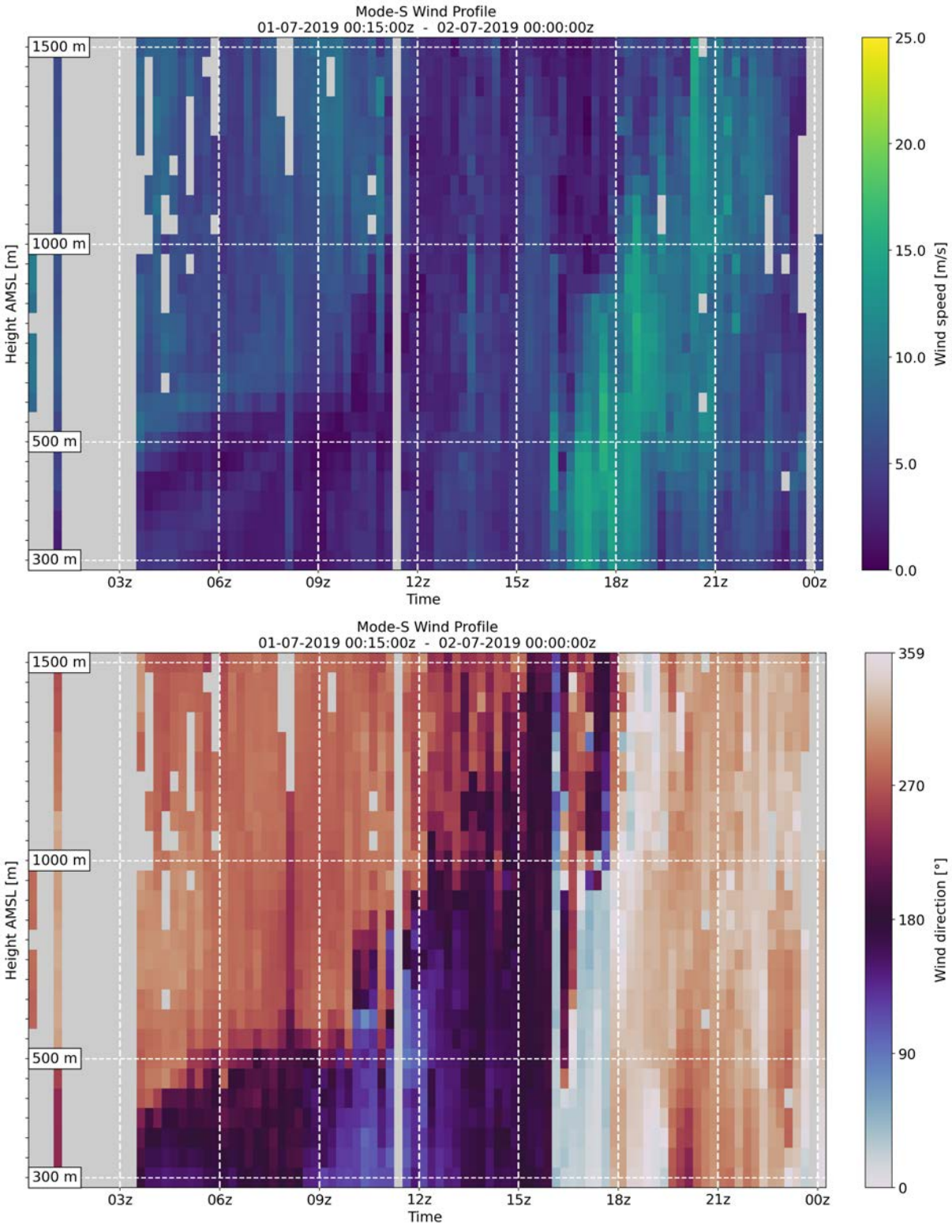


Figure 3: As Fig 2 but for reduced heights.

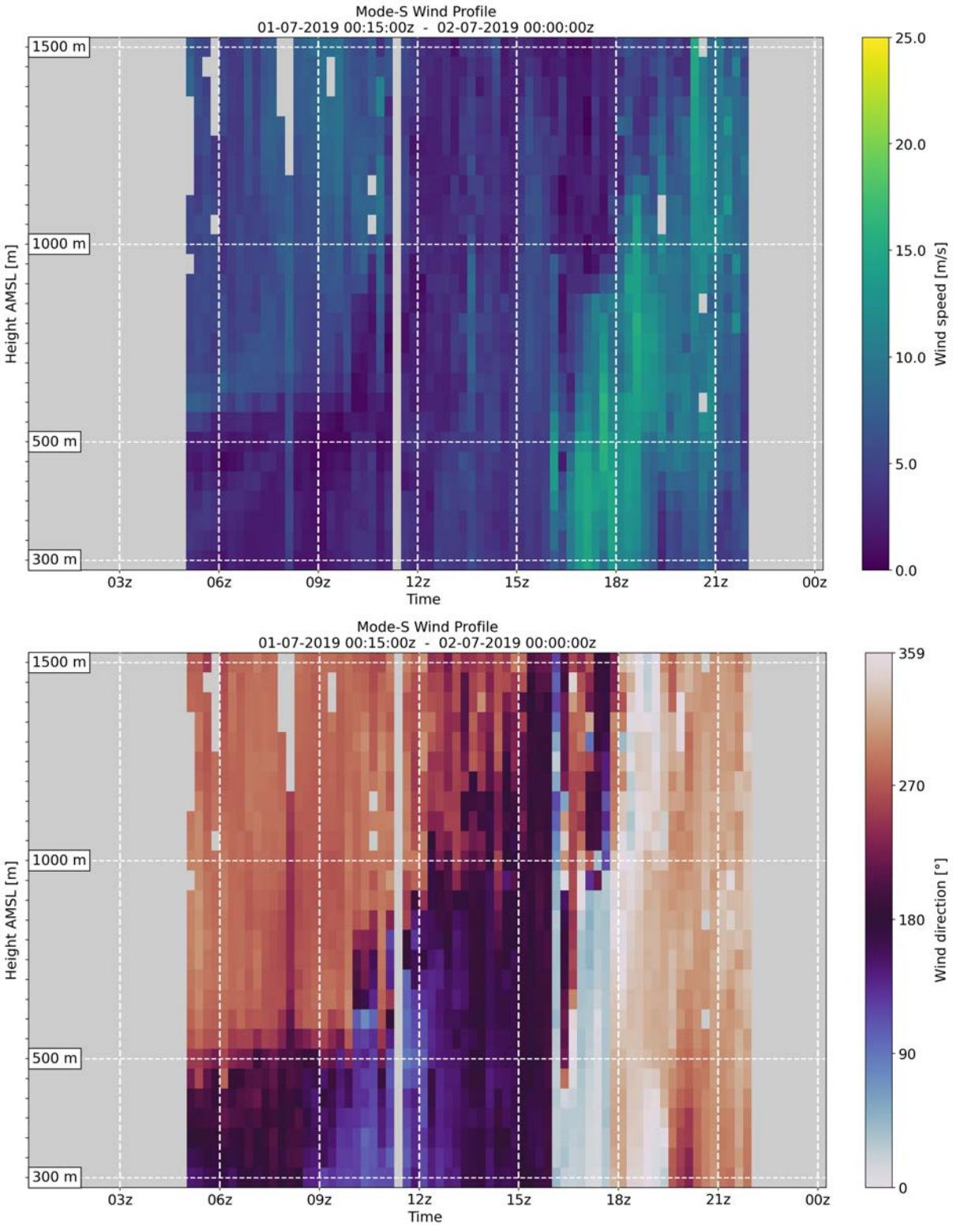
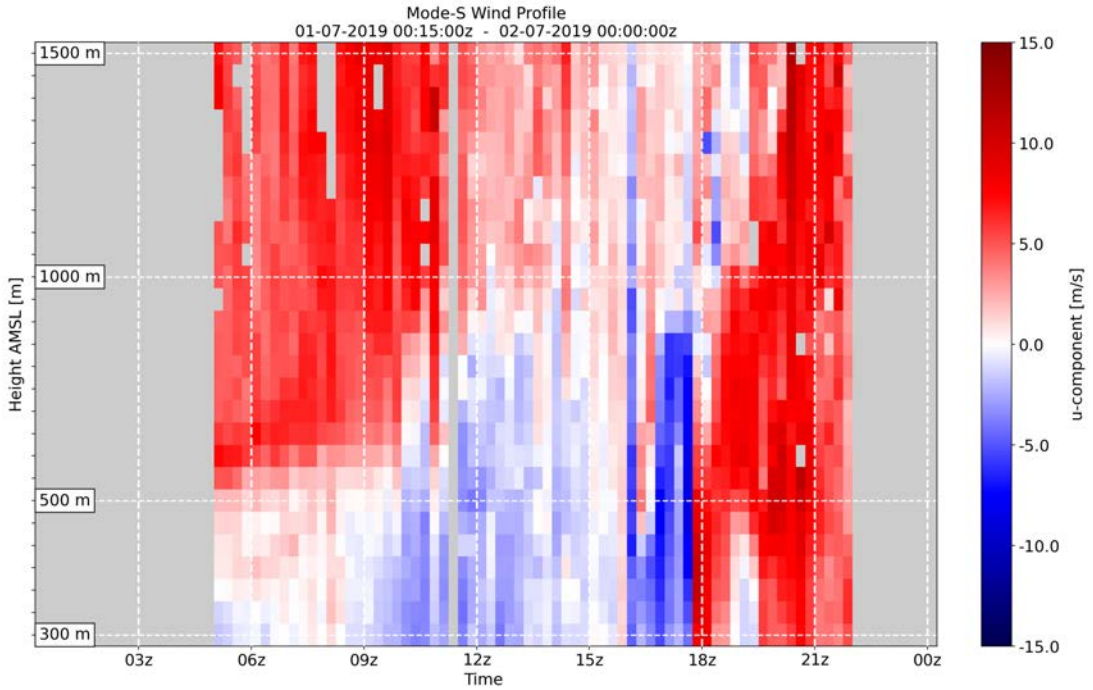
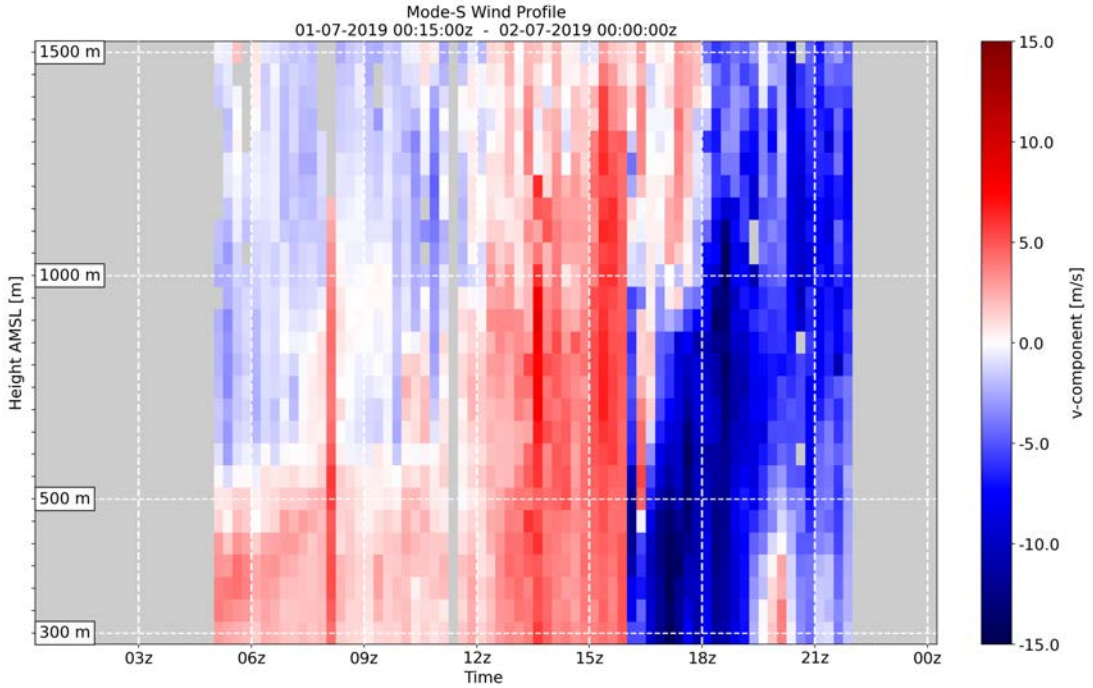


Figure 4: As Fig 3 but for reduced time coverage of wind speed ff (upper) and wind direction dd (lower).

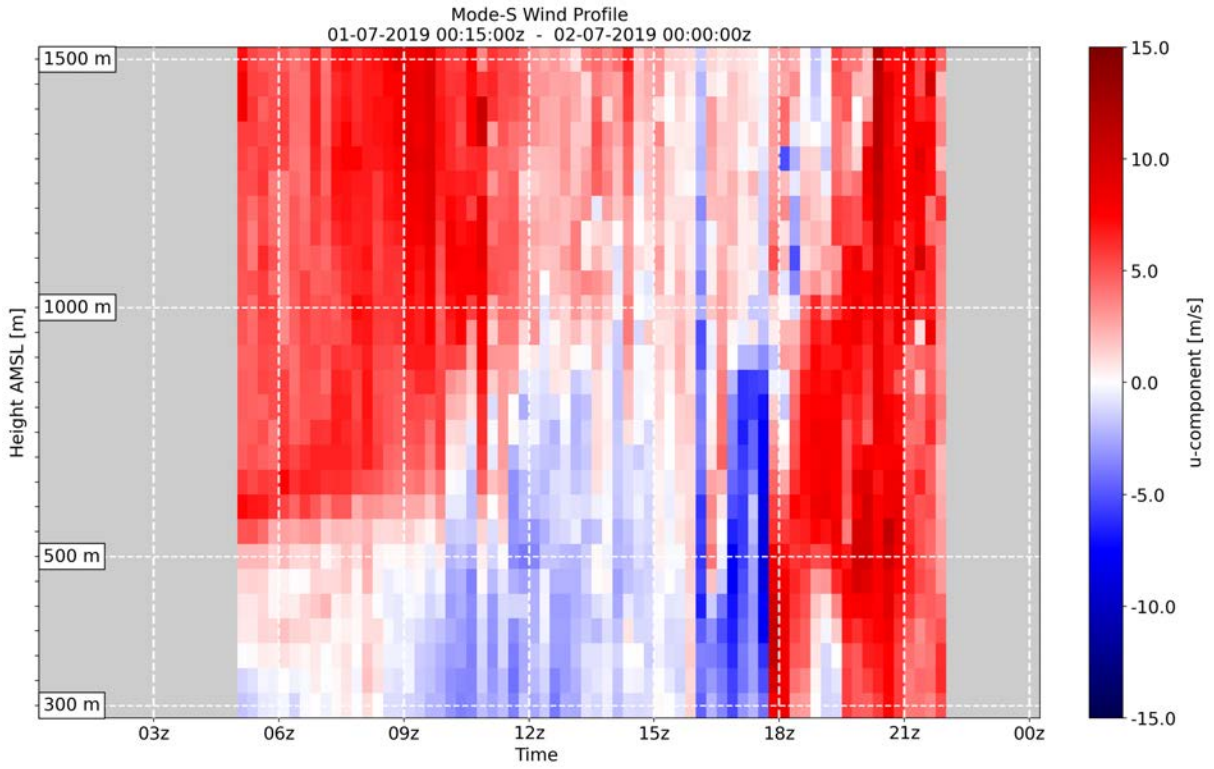


(a) u-component

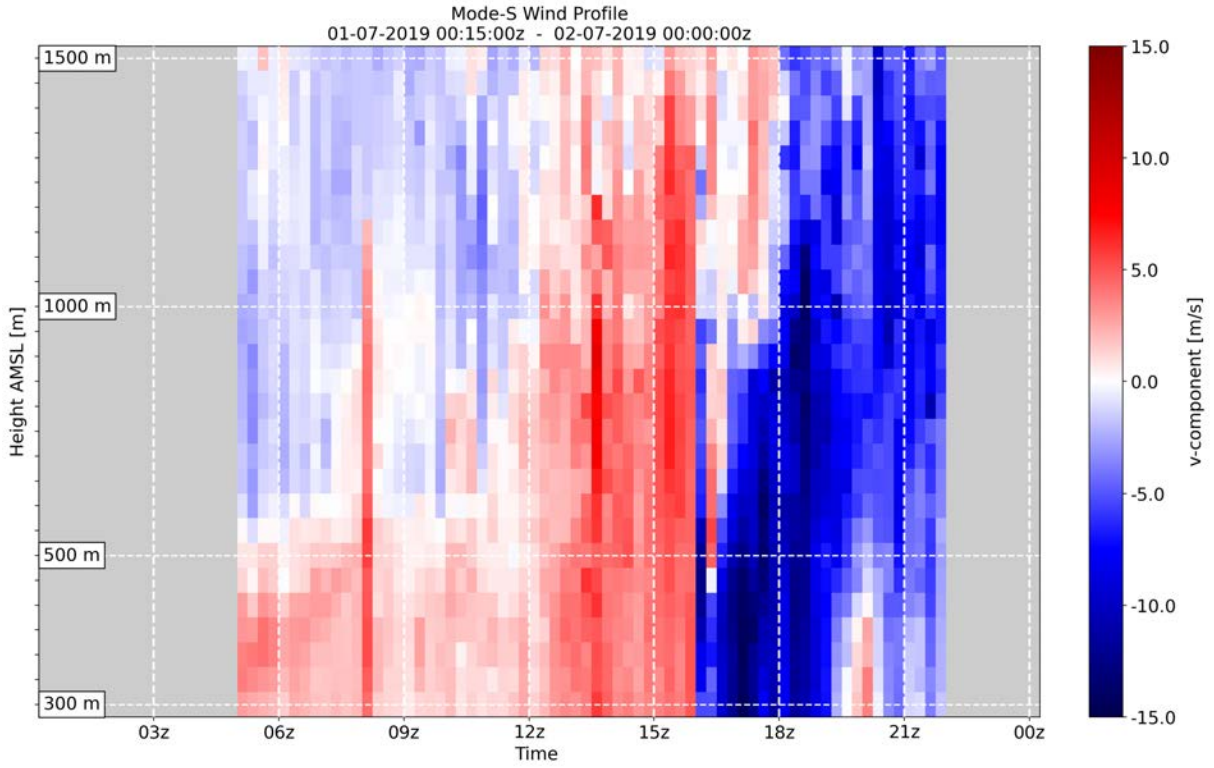


(b) v-component

Figure 5: Dataset converted into u & v component.



(a) u-component



(b) v-component

Figure 6: u & v component after filling the missing values by using median filter with a window size of 4.

3. Methods

Following comprehensive preparation of wind data to address all critical factors for AI system development, the next phase is to choose an appropriate AI algorithm for Nowcasting outputs. However, generating outputs with an AI model alone does not ensure their usefulness. Thus, a strong verification method is required to compare AI forecasts with real observations, ensuring the reliability and accuracy of the results. Figure 7 shows an overview of the task.

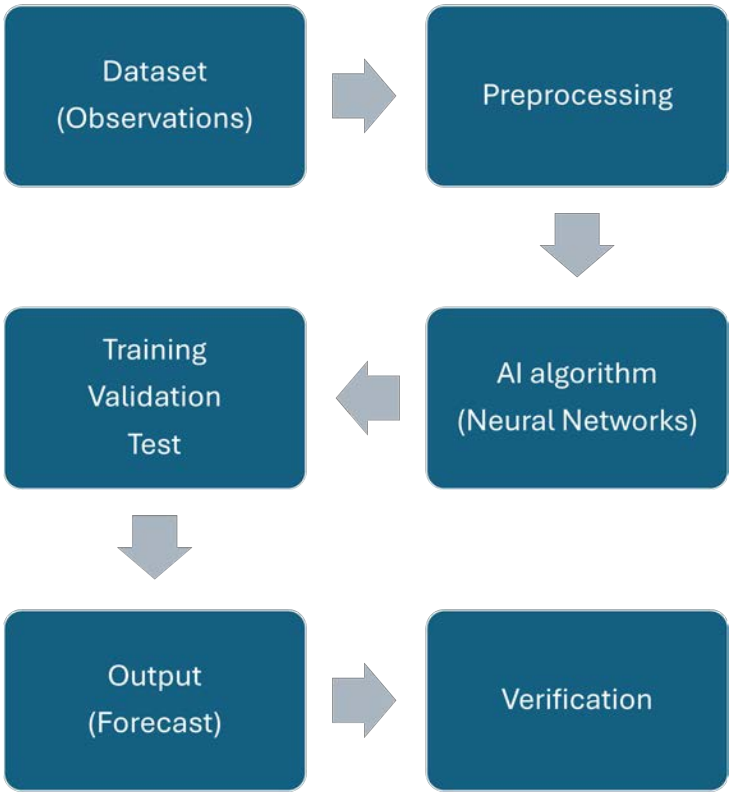


Figure 7: Overview of the machine learning process. In order to obtain a useful nowcasting of the wind, a suitable algorithm or neural network method is needed

There are several neural network models. The main ones are CNN (Convolutional Neural Network) and RNN (Recurrent Neural Network), which includes a predominant type called LSTM (Long Short-Term Memory). CNNs are frequently applied to tackle tasks related to spatial data, particularly with images, whereas RNNs are more suitable for processing sequential or time-based data (Yin et al., 2017). Since the Master thesis deals with sequential time-based datasets, RNNs are in general more appropriate to choose for training and testing the data.

3.1 Introduction

Recently, artificial intelligence (AI) has gained significant media attention. Terms like machine learning, deep learning, and AI appear frequently, even outside technical circles. Visions of a future with intelligent chatbots, autonomous vehicles, and virtual assistants are seen. For machine learning professionals, identifying real advancements from media hype is crucial, as the future of AI development depends on it. This raises key questions: What has deep learning accomplished so far? How important is it? What lies ahead? And should the hype be believed? (Chollet, 2021)
This chapter offers essential background on AI, machine learning, and deep learning. Artificial Intelligence (AI) enables machines to perform tasks that typically require human intelligence. Machine learning (ML), a subset of AI, allows systems to learn from data and improve over time. Deep learning, a branch of ML, uses multi-layered neural networks to process complex data, driving advancements in speech recognition, image analysis, and more. Figure 8 below gives an overview of the three described learning methods.

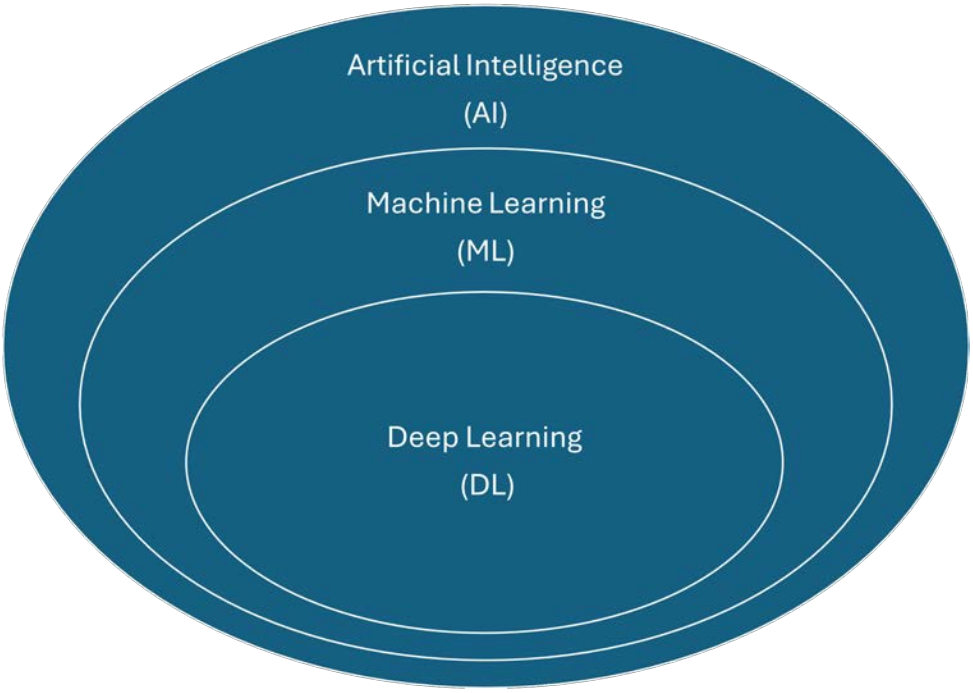


Figure 8: A general overview of the meaning of the different terms: Artificial Intelligence (AI), Machine Learning (ML) and Deep Learning (DL)

3.1.1 Artificial Intelligence (AI)

It refers to the creation and development of computer systems or machines that are capable of performing tasks that typically require human intelligence. These tasks include, but are not limited to, learning from experience, understanding and interpreting natural language, recognizing complex patterns, solving problems, and making decisions. The fundamental goal of AI is to create systems that can process information and make autonomous decisions, just as a human would, but often at a faster and more precise rate. (Chollet, 2021).

3.1.2 Machine Learning (ML)

Machine learning is a branch of artificial intelligence that enables computers to learn from data and make decisions or predictions without being explicitly programmed to do so. Instead of following predetermined rules, a machine learning model processes large sets of data, identifies patterns within that data, and adjusts its operations to improve its performance over time. This approach allows machines to "learn" from experience, similar to how humans adapt their behavior based on what they observe. (Mahesh, 2019).
Before proceeding, it is necessary to have a general overview about what machine learning algorithms do in order to understand them.
Usually, machine learning algorithms follow a general structure. The data is fed into the model as referred to as input data.
Input data is the information used to train models. It consists of features (also called attributes or variables) that represent the characteristics of the data points, and labels (in supervised learning*) that indicate the target outcome. The quality of input data significantly affects the model's performance, making data preprocessing and feature selection crucial steps in the machine learning pipeline.
After the input data passes through layers that add weights and biases to the values, it comes out as output data. In supervised learning, this includes the predicted labels or values that correspond to the input features. In unsupervised learning*, output data might involve clusters or groups identified within the input data. The accuracy and relevance of the output are key indicators of a model's effectiveness.
It is also advantageous to evaluate the model's performance. This can be done by analyzing metrics like accuracy, precision, recall, and F1 score, which provide insights into how well the model makes predictions. Overall, a well-performing algorithm should generalize well to unseen data, accurately capturing patterns without being overly complex or too simplistic. Effective evaluation is crucial for refining models and ensuring they meet the desired objectives.

3.1.3 Deep Learning (DL)

Deep learning is a branch of machine learning that uses artificial neural networks to model and understand complex patterns in data. These neural networks consist of multiple layers that transform data through a process of learning, similar to how the human brain functions (Chollet, 2021).
The key components in deep learning are the neural network architecture, learning process, and training. A DL model consists of layers of interconnected nodes, also known as neurons. Each neuron processes input data and passes the result to the next layer. Those layers are known as the input layer, which receives the data, hidden layer, where computations and transformations are made, and the output layer, where the final results are produced.
The learning process includes forward propagation, where the input data is passed through the layers, a loss function that calculates the error or the loss, and backpropagation, where the error will be reduced through weights adjustment (e.g. using the gradient descent method).
Training the model is also an important key component. During each iteration (epoch), the weights are updated based on the feedback from the loss function. In this case, the accuracy of the model can be improved over time.

*refer to page X for more details about supervised and unsupervised learning

Neural networks can have a simple or complex structure. The difference between the simple one (shallow network) and the complex one (deep neural network) is described in figure 9 and 10 below. A shallow network contains only one simple hidden layer, and a deep neural network (DNN) contains at least two hidden layers.

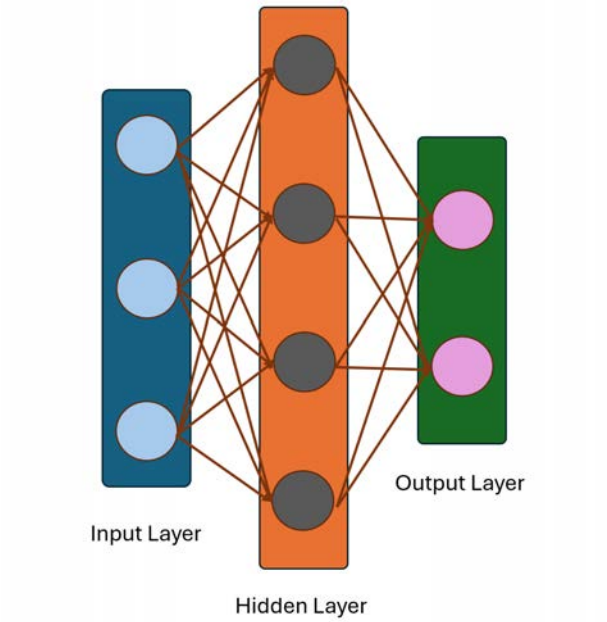


Figure 9: A shallow network, where the input data passes through one simple hidden layer and exit through the output layer

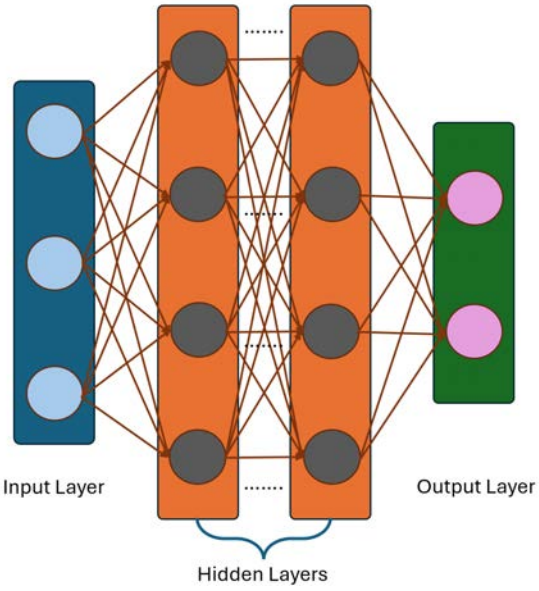


Figure 10: A deep neural network (DNN), where the input data passes through multiple hidden layers and exit through the output layer

In deep learning, models are designed to handle different types of tasks depending on the data and the objective. The tasks generally fall into three broad categories: supervised learning, unsupervised learning, and reinforcement learning. Each type is used in different contexts depending on the case of the problem and will be described further below.

Supervised Learning:

In supervised learning, the model learns from labeled data, where both the input and the desired output (label) are provided. The goal is to learn a mapping from inputs to outputs (Sarker, 2021). Examples of tasks for supervised learning are classification, which refers to categorizing data into predefined classes (e.g., image classification, spam detection) and regression, which refers to predicting sequential data or values (e.g., stock prices prediction, weather data prediction).
Two main algorithms of supervised learning are CNN and RNN. This type of deep learning technique is used in this Master thesis and will be further discussed in section 3.2 below.

Unsupervised Learning:

Unsupervised learning is a type of machine learning in which a model is trained on unlabeled data, meaning that it does not have predefined output. The goal is to discover patterns, relationships, or structures within the data without explicit guidance. Unlike supervised learning, where the model learns from input-output pairs, unsupervised learning identifies hidden features based solely on the data itself (Sarker, 2021). Example functions are clustering, where data points are grouped into clusters based on similarity (e.g., customer segmentation) and dimensionality reduction, as it is about reducing the number of input features while retaining important information (e.g., PCA, t-SNE).

Reinforcement Learning:

The third type of machine learning is reinforcement learning, where an agent learns to make decisions by interacting with an environment. The agent takes actions based on the current state, and the environment provides feedback in the form of rewards or penalties. The goal is to maximize cumulative rewards over time by learning the best strategy to make a decision (Sarker, 2021). Some examples are game playing, robotics, and real-world tasks like self-driving cars. The agent takes actions in an environment, receives feedback, and adjusts its actions to maximize the reward over time.

3.2 Algorithms

Since this thesis deals with sequential time series datasets of wind profiles, the suitable algorithm for predicting those datasets would be the Recurrent Neural Network (RNN), because RNNs have a special architecture that allows them to retain memory of previous inputs, while a normal feed-forward neural network has the input data basically pass through the model without retaining memory of previous inputs.
Basic RNNs usually have issues (will be discussed later), e.g. when it comes to long sequence datasets; therefore, two improved versions of RNN have been developed to handle this problem: Long-Short Term Memory (LSTM) and Gated Recurrent Unit (GRU). The difference between LSTM and GRU is that GRU has a simplified structure and fewer parameters than LSTM.
Due to the long time series datasets of the wind profiles, the improved version LSTM will be the most appropriate to use in this case of study.

3.2.1 Feed-Forward Neural Network (FFNN)

Starting with the basic neural network architecture, this kind of neural networks has no cycle between the nodes. The reason for calling it "$Feed-Forward$" is due to the direct one-way pass of the information. It starts with the input layer, passes through the hidden layers (if they exist) and lastly to the output layer. It doesn't include any loops or feedback connections; that's the reason why it doesn't retain information from previous inputs (Sazli, 2006).
Figure 11 shows a general structure of a FFNN, where the input layer consists of nodes. Each node represents an input feature $\mathbf{x}_n$, where n is the number of samples. Those nodes let the input information pass through to the upcoming layer. This upcoming layer is defined as the hidden layer $\mathbf{h}_n$, where it can contain multiple layers, sometimes even none. It depends on the needs. Each hidden layer contains many neurons. Those neurons apply weights ($\mathbf{W}_x$, $\mathbf{W}_y$) and biases ($\mathbf{b}_h$, $\mathbf{b}_y$) to the inputs and process them using activation functions ϕ_h and ϕ_y, where the activation functions are typically a logistic sigmoid, tanh function, or Rectified Linear Unit (ReLU). After that, the information will pass further to the next layer. The output layer is the last step in the process, where the output features $\mathbf{y}_n$ will be introduced. How many neurons the output layer contains depends on the type of the problem. All the notations before can be put together and result in the hidden and output variable equations:

$$\mathbf{h}_n = \phi_h(\mathbf{x}_n\mathbf{W}_x + \mathbf{b}_h) \tag{5}$$

$$\mathbf{y}_n = \phi_y(\mathbf{h}_n\mathbf{W}_y + \mathbf{b}_y) \tag{6}$$

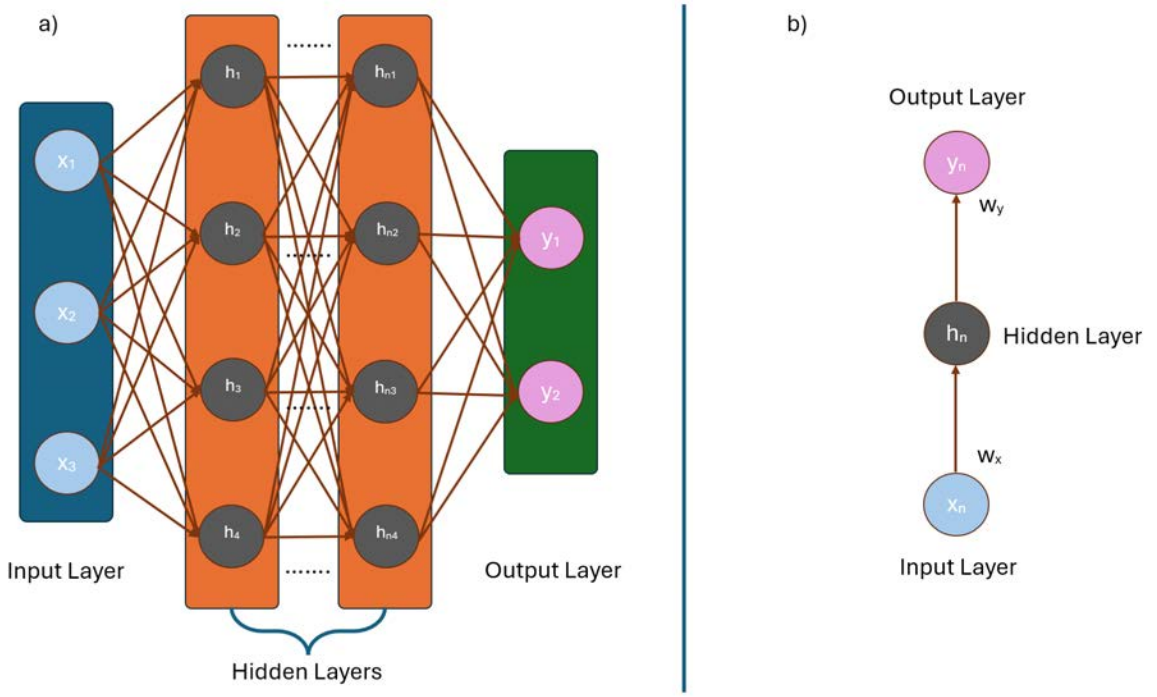


Figure 11: FFNN structure where a) represents the detailed feed-forward neural network with all nodes, layers and its connections and b) represents a general overview structure of FFNN.

3.2.2 Recurrent Neural Network (RNN)

RNN is a type of supervised learning where the model learns from labeled data. Its main use is to identify patterns within sequential data. Unlike feed-forward neural networks, RNNs incorporate cyclic connections, which enable them to retain information from earlier inputs (Schmidt, 2019). Similar to FFNN, Figure 12 shows the generic structure of an RNN with some differences.
The only difference to FFNN is that RNN has cyclic connections, that means, each node of the hidden layer does not pass through the layers one way. It allows the information to return to the network again.

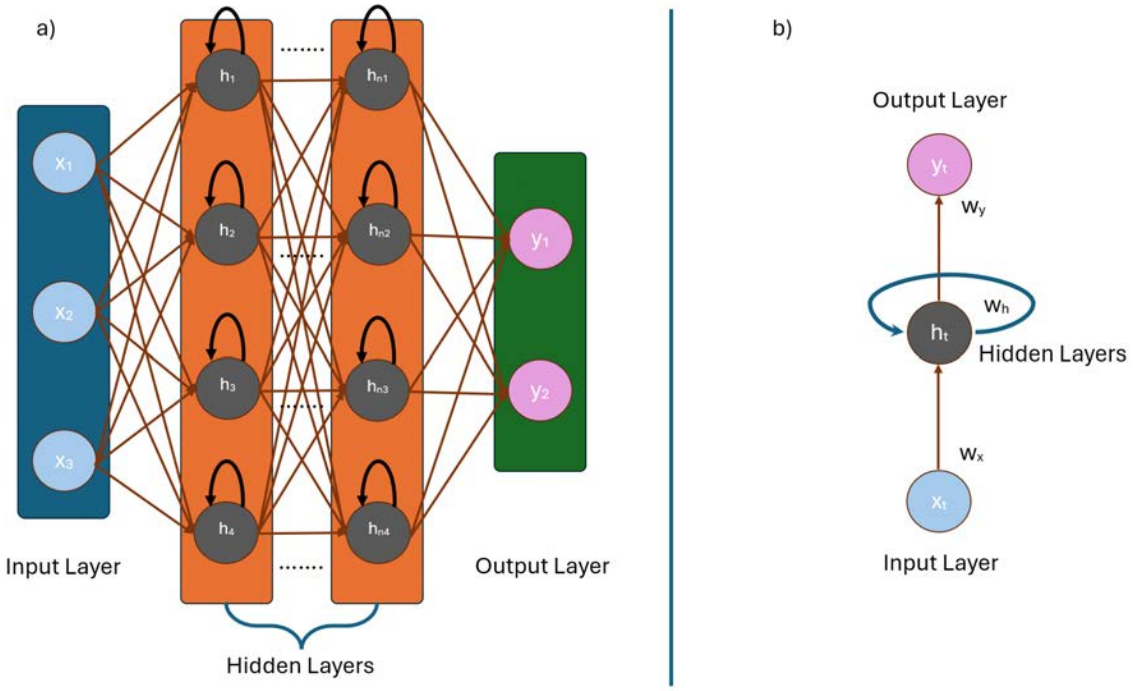


Figure 12: RNN structure where a) represents a detailed RNN with all nodes, layers and its cyclic connections in the hidden layer and b) is a general overview structure of RNN.

Due to the cyclic connections also called feedback loops, the output of each previous time step is used as an input for the current time step. Hence, RNN has a form of memory allowing it to be appropriate for sequential datasets. Figure 13 shows how a fully connected RNN with time steps works.
This extension of the FFNN also leads to some notation differences from the one before. Equation 5 will be modified by adding only a new weight to the hidden state $\mathbf{W}_h$ of each previous time step as follows:

$$\mathbf{h}_t = \phi_h(\mathbf{x}_t\mathbf{W}_x + \mathbf{h}_{t-1}\mathbf{W}_h + \mathbf{b}_h) \tag{7}$$

Equation 6 will remain similar, but it will only be time dependent:

$$\mathbf{y}_t = \phi_y(\mathbf{h}_t\mathbf{W}_y + \mathbf{b}_y) \tag{8}$$

It is also necessary and crucial to perform backpropagation of the error through time (BPTT) during training RNN to learn from sequential data by capturing temporal dependencies. RNN can have computing problems (e.g. vanishing and exploding gradients) when having a very large number of sequential data. This can be approved by showing some mathematical notation presented by (Schmidt, 2019) in the next page.

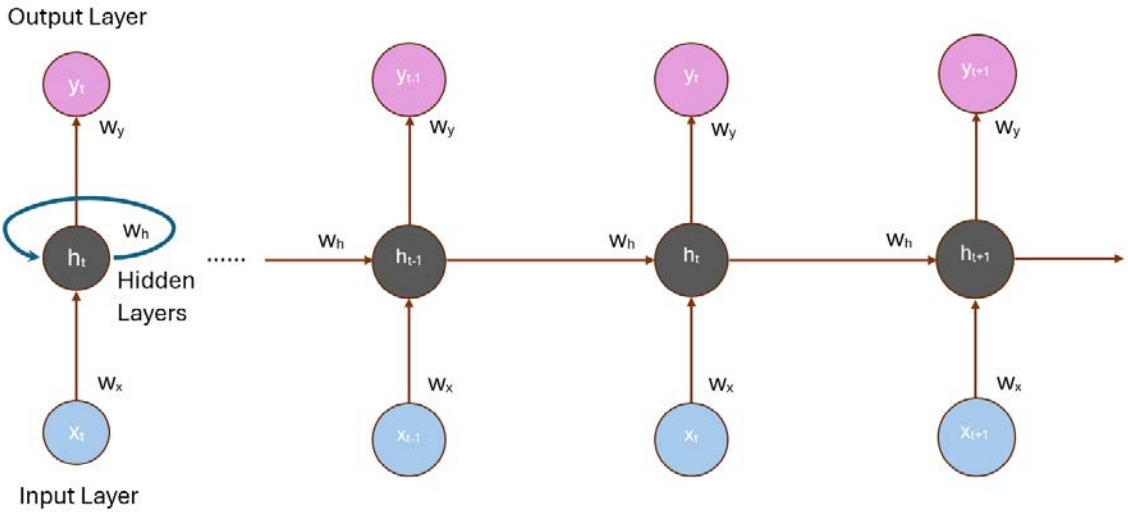


Figure 13: Fully connected RNN with hidden layers time steps.

Backpropagation Through Time (BPTT):

BPTT is a method specially made to train RNNs to handle time dependencies. It begins by unrolling the RNN across all time steps, and the weights are shared across the layers. While the forward pass computes the hidden state for each time step, the current hidden state will be updated from the current input and the hidden state of the previous time step. The output will be determined from each hidden state at each time step.
There is also a backward pass, where the gradients of the loss will be calculated across all previous time steps by backpropagating the error through the unrolled network, taking into account how previous states affect the current state. After calculating the gradients, the weights are updated using gradient descent (or some variant), similar to other neural networks.
Deploying a Loss function $\mathbf{L}(\mathbf{y}_n, \hat{\mathbf{y}}_n)$ in BPTT (eq. 9) is necessary to determine the difference between the predicted output values and the true output values over the time steps.

$$\mathbf{L}(\mathbf{y}_n, \hat{\mathbf{y}}_n) = \sum_{t=1}^{T} \ell_t(\mathbf{y}_t, \hat{\mathbf{y}}_t) \tag{9}$$

Where ℓ_t is the loss function at each time step. ℓ_t can have multiple definitions depending on the task, e.g. by having regression tasks ℓ_t would be the Mean Squared Error (MSE) or for classification tasks the Cross-Entropy Loss.

Now calculating the partial derivative of each of the three weights we defined earlier ($\mathbf{W}_x, \mathbf{W}_h, \mathbf{W}_y$) by using the chain rule we will obtain the following results respectively:

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_y} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \frac{\partial \phi_y}{\mathbf{W}_y} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \mathbf{h}_t \tag{10}$$

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_h} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \frac{\partial \phi_y}{\partial \mathbf{h}_t} \cdot \frac{\partial \mathbf{h}_t}{\partial \phi_h} \cdot \frac{\partial \phi_h}{\partial \mathbf{W}_h} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \mathbf{W}_y \cdot \frac{\partial \mathbf{h}_t}{\partial \phi_h} \cdot \frac{\partial \phi_h}{\partial \mathbf{W}_h} \tag{11}$$

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_x} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \frac{\partial \phi_y}{\partial \mathbf{h}_t} \cdot \frac{\partial \mathbf{h}_t}{\partial \phi_h} \cdot \frac{\partial \phi_h}{\partial \mathbf{W}_x} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \mathbf{W}_y \cdot \frac{\partial \mathbf{h}_t}{\partial \phi_h} \cdot \frac{\partial \phi_h}{\partial \mathbf{W}_x} \tag{12}$$

Due to the relation of $\mathbf{h}_t$ on the previous time steps, substituting the last part of the equations 11 and 12 will result in new equations:

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_h} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \mathbf{W}_y \sum_{k=1}^{t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} \cdot \frac{\partial \mathbf{h}_k}{\partial \mathbf{W}_h} \tag{13}$$

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_x} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \mathbf{W}_y \sum_{k=1}^{t} \frac{\partial \mathbf{h}_t}{\partial \mathbf{h}_k} \cdot \frac{\partial \mathbf{h}_k}{\partial \mathbf{W}_x} \tag{14}$$

Now rewriting equation 13 and 14 yields:

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_h} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \mathbf{W}_y \sum_{k=1}^{t} (\mathbf{W}_h^{\top})^{t-k} \cdot \mathbf{h}_k \tag{15}$$

$$\frac{\partial \mathbf{L}}{\partial \mathbf{W}_x} = \sum_{t=1}^{T} \frac{\partial \ell_t}{\partial \mathbf{y}_t} \cdot \frac{\partial \mathbf{y}_t}{\partial \phi_y} \cdot \mathbf{W}_y \sum_{k=1}^{t} (\mathbf{W}_h^{\top})^{t-k} \cdot \mathbf{x}_k \tag{16}$$

Due to the large weights of the hidden layer $\mathbf{W}_h$ in equations 15 and 16. It might result in a very large computational loss function **L**. Hence, the gradients of the loss function **L** can either become extremely small (vanish) or extremely large (explode) as they are propagated backward through the time steps. An appropriate solution for this issue is to use the Truncated BPTT. For this approach the sequence will be divided into smaller segments and the loss is computed and backpropagated over these shorter segments.

Why do gradients vanish or explode?

As shown in the previous equations, the derivative of the hidden state $\mathbf{h}_t$ depends on the derivative of the previous hidden state $\mathbf{h}_{t-1}$. It is just a multiplication of terms for each time step. If the derivative of the activation function ϕ at each time step is less than 1, the gradient in this case will decrease step by step until it vanishes.
If the derivative of the activation function ϕ or weights **W** at each time step is larger than 1, then the gradient in this case will increase step by step until it becomes numerically unstable and explodes.

A lot of attempts have been developed to get rid of this problem. From those attempts are, for example, using different activation functions like ReLU (Rectified Linear Unit) to reduce the problem of vanishing. The derivative of such an activation function is either 0 or 1. Hence, small gradients will be avoided.
Another method is the gradient clipping. Gradient clipping involves setting a threshold value for the gradients. If the computed gradient exceeds this threshold, it is scaled down to the threshold value, and it keeps the gradients from getting larger and larger.
Two other most interesting and important methods in this study are the Long-short Term Memory (LSTM) and Gated Recurrent Unit (GRU). LSTM and GRU are a special type of RNNs, and they use gates in their architecture to control the flow of information. In this master thesis, the LSTM algorithm will be used and discussed explicitly in the next section.

3.2.3 Long-Short Term Memory (LSTM)

LSTM is a special type of RNN that deals with the problem of vanishing gradient. It uses a set of gates to regulate the flow of information through the network, enabling it to maintain and manipulate memory over longer sequences (Hochreiter and Schmidhuber, 1997).
The concept of LSTM is similar to RNN before, but the structure is quite different within a time step. As shown in figure 13 each time step depends on the previous ones and weights and biases are considered at each time step. For a deeper comparison between LSTM and RNN, figure 14 shows a similar type of structure to figure 13, but this time as a cell with tanh as an activation function.

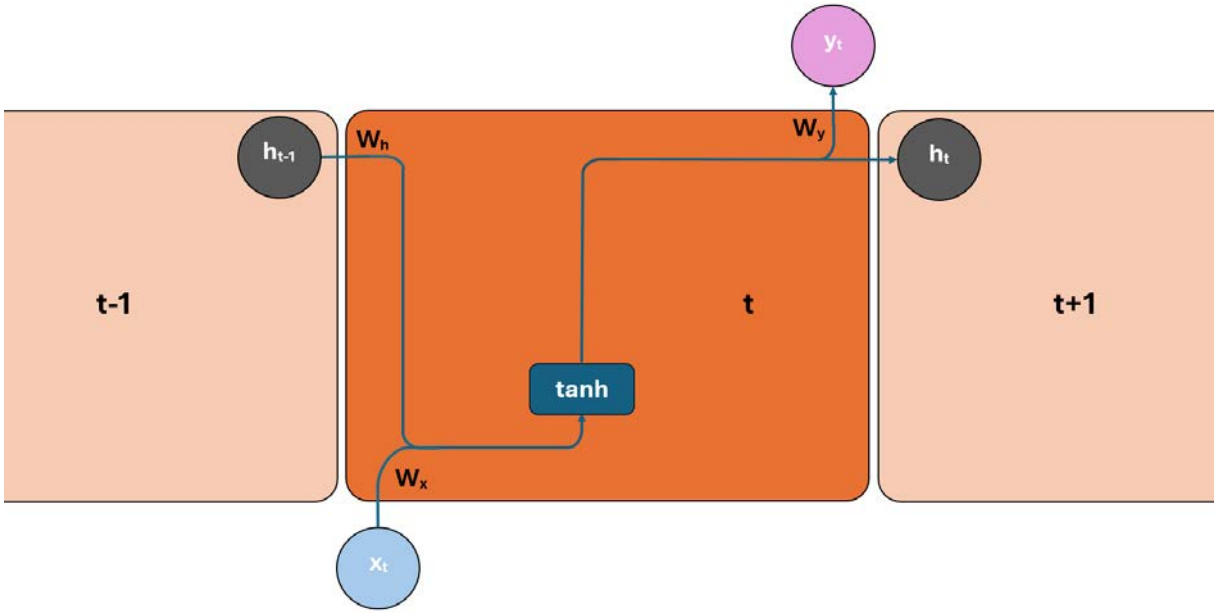


Figure 14: RNN cell structure

LSTM has in each time step three gates (see fig. 15); the first one is the forget gate $\mathbf{f}_t$. It determines which information should be ignored or not. Both the current input state $\mathbf{x}_t$ and the previous hidden state $\mathbf{h}_{t-1}$ will be taken into account and pass through a sigmoid activation function (σ). The result of this gate is a value of 0 or 1, where 0 is to forget the information and 1 to keep the information for the next time steps.

$$\mathbf{f}_t = \sigma(\mathbf{x}_t\mathbf{W}_{xf} + \mathbf{h}_{t-1}\mathbf{W}_{hf} + \mathbf{b}_f) \tag{17}$$

The second gate is the input gate $\mathbf{i}_t$. It tells which new information shall enter the cell. If the information has entered the cell, it will have to pass through 2 layers; the first is the sigmoid layer, which is represented mathematically in eq. 18. It decides which value to be updated. Second is the tanh layer (eq. 19), which generates a vector of new candidate values (candidate memory) $\tilde{C}_t$ that can be added to the cell state.

$$\mathbf{i}_t = \sigma(\mathbf{x}_t\mathbf{W}_{xi} + \mathbf{h}_{t-1}\mathbf{W}_{hi} + \mathbf{b}_i) \tag{18}$$

$$\tilde{C}_t = tanh(\mathbf{x}_t\mathbf{W}_{xc} + \mathbf{h}_{t-1}\mathbf{W}_{hc} + \mathbf{b}_c) \tag{19}$$

Eq. 20 shows the mathematical notation of the output gate $\mathbf{o}_t$, which determines what the next hidden state $\mathbf{h}_t$ (eq. 21) will be. The cell state is passed through a tanh activation function and then multiplied by the output gate's sigmoid output (σ).

$$\mathbf{o}_t = \sigma(\mathbf{x}_t \mathbf{W}_{xo} + \mathbf{h}_{t-1} \mathbf{W}_{ho} + \mathbf{b}_o) \tag{20}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot tanh(\mathbf{C}_t) \tag{21}$$

Where $\mathbf{C}_t$ is the updated cell state of the new information and the previous state and defined in eq. 22.

$$\mathbf{C}_t = \mathbf{f}_t \odot \mathbf{C}_{t-1} + \mathbf{i}_t \odot \tilde{C}_t \tag{22}$$

The advantage of these gates is the forgetting and retaining of information at each new time step, which gives control of the information flow and helps in learning long-term dependencies in sequential data.

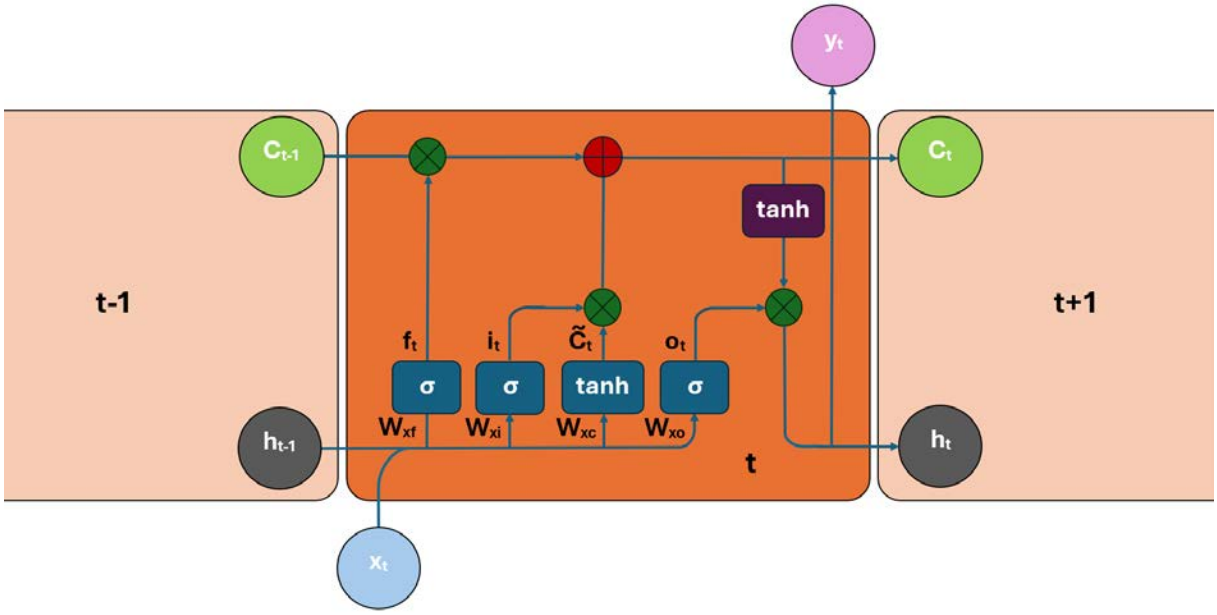


Figure 15: LSTM cell structure. The $\otimes$ represents vector multiplication and the $\oplus$ vector addition.

3.3 Verification

After predicting the wind values using the model, it is necessary to verify and compare the output values with the actual observations to check whether they are useful or not. There are multiple common verification methods to use. For example, statistical metrics to evaluate the accuracy and reliability of the model predictions, like Mean Absolute Error (MAE), Root Mean Square Error (RMSE), etc.;
visualization methods that give a clear understanding of the effectiveness of the model through residual plots, scatter plots, etc.;
and skill scores that benchmark the model against baseline models, e.g., the Brier score and Index of Agreement (IoA).
Applying all these methods will help the output results of wind speed and direction be compared and evaluated, in order to observe whether they are useful for the aim of ATM.

3.3.1 Statistical metrics

MAE and RMSE are two statistical metrics that are used to measure the error in the prediction error of the model and to evaluate the model (Hodson, 2022). There are some key differences between MAE and RMSE. The main difference is that MAE calculates the average of absolute errors, while RMSE calculates the square root of mean squared errors. MAE is therefore less sensitive to outliers than RMSE. The choice of which of the two to use depends on how the dataset is constructed and what kind of values it contains. It is recommended to use MAE if the dataset has outliers or extreme values, and it is only suitable to calculate for the wind speed as in equation (23). If the errors are critical and need a higher penalty, then it is recommended to use RMSE as defined in equation (24).

$$MAE_{ff} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{23}$$

$$RMSE_{ff} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{24}$$

Where y_i and $\hat{y}_i$ are the observations and predictions, respectively. MAE and RMSE for wind direction will be calculated as suggested by Jiménez and Dudhia (2013):

$$MAE_{dd} = \frac{1}{n}\sum_{i=1}^{n}|\Delta dd_i| \tag{25}$$

$$RMSE_{dd} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\Delta dd_i)^2} \tag{26}$$

Where the difference Δdd_i between predictions and observations is given as:

$$\Delta dd_i = \begin{cases} \hat{y}_i - y_i & if \quad \hat{y}_i - y_i \leq |180| \\ \hat{y}_i - y_i - 360 & if \quad \hat{y}_i - y_i > 180 \\ \hat{y}_i - y_i + 360 & if \quad \hat{y}_i - y_i < -180 \end{cases}$$

Since the dataset has outliers and the errors need a higher penalty, both metrics will be used and presented later in the results.

There is also another useful metric similar to MAE called MAPE (Mean Absolute Percentage Error), which measures the average percentage difference between the observations and predictions (Eq. 27). It is also used to assess model accuracy and is only reliable with the wind speed (Kim and Kim, 2016).

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \times 100 \tag{27}$$

If MAPE = 0 % then it is a perfect prediction and has no errors. A MAPE value of < 10 % can indicate an excellent prediction, while a MAPE between 10 and 20 % can have good accuracy and between 20 and 50 % moderate accuracy. Any MAPE value below 50 % indicates poor accuracy, and the model should be further improved.

Another useful statistical measure suggested to be used by Chicco et al. (2021) is the coefficient of determination (R^2). This coefficient can tell how good the fit of the regression is between actual and predicted data. It is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y_i})^2} \tag{28}$$

Where $R^2 = 1$ means excellent fit, $R^2 = 0$ indicates that the model doesn't explain any of the variance, it is simply predicting the mean, and $R^2 < 0$ the model is performing worse than predicting the mean.

3.3.2 Skill scores

A skill score is used for the evaluation of the performance of a forecast model relative to a baseline. It reveals whether the forecast model is better or worse compared to the baseline. Usually, skill scores are normalized, and they are defined with a range from -∞ to 1, where scores with negative values represent that the model performs worse than the baseline, a score of 0 means that the model performs no better than the baseline, and a score of 1 is a perfect model (Wheatcroft, 2019).
The following equation represents the general one for a skill score (SS):

$$SS = 1 - \frac{E_{Model}}{E_{Baseline}} \tag{29}$$

Where, E_{Model} is the error of the forecast model and the $E_{Baseline}$ is the error of the reference model. Both of those errors depend on the metrics used in the case of the study; they could be metrics like, for example: MAE, MSE, or RMSE.
Skill scores are usually useful in meteorology and specifically in time series forecasting. Normally, they provide a clear comparison against the baseline, and they give a simple and single value about the effectiveness of the model.
There is another skill score called Index of Agreement (IoA). It is also a measure of evaluation of the accuracy of predictive models by comparing their outputs to actual data. It has a range of scores between 0 and 1, where the score 0 indicates that the outputs do not agree with the observations, and the score 1 means excellent agreement. The calculation formula of IoA regarding Huang et al. (2021) is defined as the following:

$$IoA = 1 - \frac{\sum_{i=1}^{n}(P_i - O_i)^2}{\sum_{i=1}^{n}(|P_i - \overline{O}| + |O_i - \overline{O}|)^2} \tag{30}$$

Where P_i are the predicted values, O_i represents the actual values, $\overline{O}$ is the mean of the actual data, and n is the number of observations.

4. Results

After having a complete imputed dataset from Section 2.2, the next step is to determine appropriate sizes for the input and output window. Due to the limited availability of the dataset between 5 to 22 UTC, the input window is ideally 6 hours (=24 time steps). Choosing an input window longer than 6 hours will result in higher RMSE values between the original data and the predicted ones. Choosing an input window of less than 6 hours will keep the RMSE relatively stable, but the model will not have enough input data per window or instance, which is insufficient for model training. The output window (prediction) is 2 hours, and this would fulfill the question of the research (nowcasting) and the need of ATM.
In the next sections, we will get into the details of the model, comparisons, and choice of the results.

4.1 Model structure and effectiveness

After splitting the data into training, validation, and test sets for both u- and v-components, a window needs to be generated to define the base of the model. This defined window consists of multiple parameters: the input width, which describes the number of time steps used for the training of the LSTM model; the label width (output width), which tells the number of time steps that the model expects to predict; and the shift, which defines how much space is between the end of the input sequence and the beginning of the label sequence. All these parameters can be defined and set as needed. For example, if the requested input is 6 hours and the output is 2 hours, the input and label width would be 24 and 8, respectively, due to the 15 min interval of each time step. The shift would also be ideally 8, so that no time steps from the input sequence are trained repeatedly in the next forward windows. Additional parameters are to be selected. e.g., the batch size (or so-called batching operation). It is used to converge the elements of a dataset, e.g. by combining the inputs and labels in a single batch. Batching also does shape transformation, i.e. it changes the shape of the data. Lastly, it also does some optimization for the training; it can increase the computational efficiency and reduce the noise in gradient updates. The size of the batch can vary from model to model. It is usually a geometric sequence (e.g. 16,32,64,128,....etc). Larger batch sizes typically demand more memory and can accelerate convergence; however, they may lead the optimization process to settle in less optimal solutions, rather than reaching the best possible outcome (suboptimal minima). Smaller batches can add noise to the gradient calculations, which can leave the suboptimal minima, but might converge slower. The ideal batch size to be used is 512, due to the good effectiveness and improvement of the performance.
Another important parameter is the number of epochs. It determines the maximum number of complete passes through the whole training dataset during the training. It also improves performance on the training data and enhances the weights of the model. Meanwhile, increasing the number of epochs might result in overfitting, where the model performs well on the training data but poorly on unseen data, and the opposite happens when choosing a small number of epochs or using a function called early stopping can result in underfitting, where the model would not learn enough from the training. For example, using a number of epochs of 110 means that the model iterates over the whole dataset 110 times during training.

Before using the actual available data, one can test the model's effectiveness with synthetic data and observe the suitability of the results. The synthetic data can be, for example, constant values that decrease with height and have the same shape as the real data. Figure 16 shows the input data with-

out the predictions of the model, i.e. with the labels, and Figure 17 shows the input data with the predictions of the model. Both figures have the same configuration.

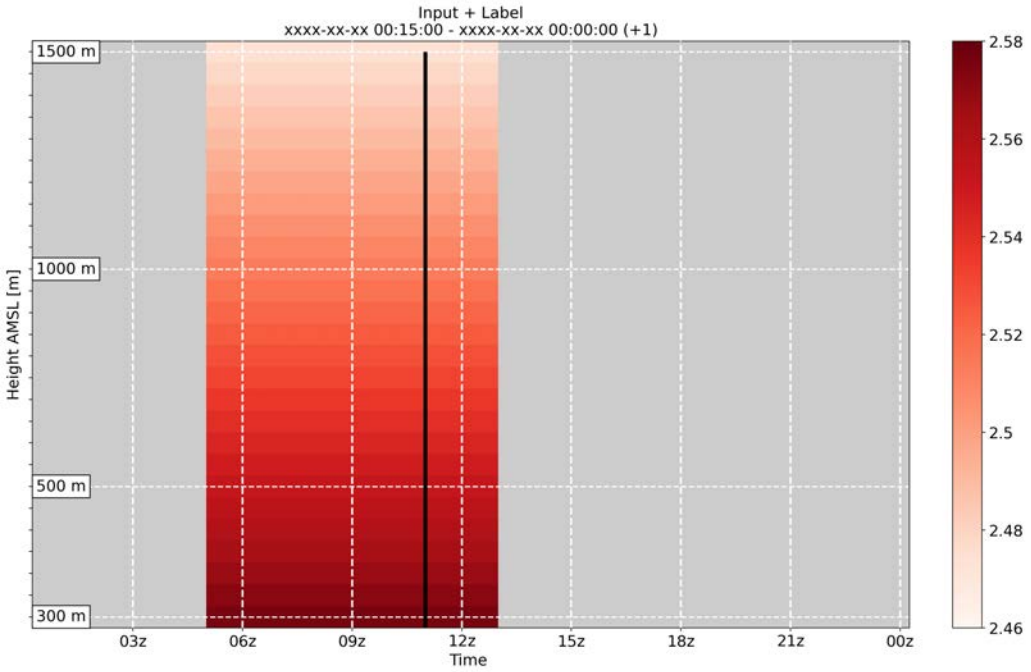


Figure 16: Synthetic data without prediction. The black vertical dividing line is to separate between the input and label/prediction data. The first 24 time steps (=6 hrs) left of the black line are the inputs and the 8 time steps (=2 hrs) right of the black line are the labels.

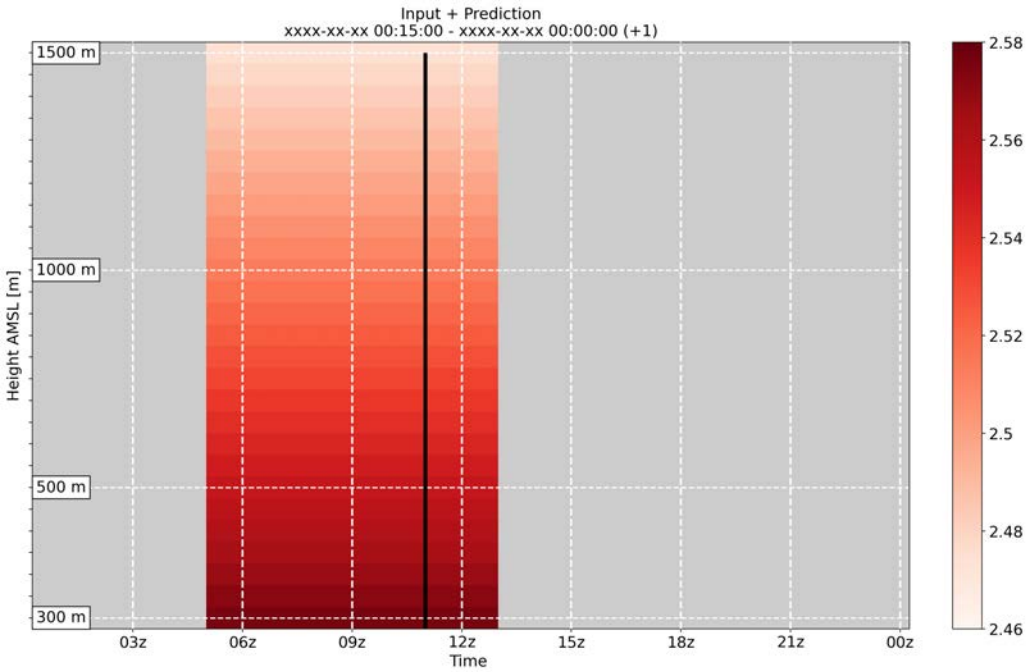


Figure 17: Synthetic data with prediction of the model.

The results of the prediction of the synthetic data indicate that the model works well and is ready to use with real data.

4.2 Model results

In section 2.2 of data preprocessing, the wind data was converted from wind speed ff and wind direction dd into u and v components, to deal only with one type of training data. Due to the complexity of reading the results as u and v components, the outputs will be converted again into wind speed ff [m/s] and wind direction dd [°].
Based on the data availability and the model parameter settings, the model is expected to show different outputs depending on the weather conditions. The model learns to predict the wind easier from unchanged weather conditions than from rapidly or suddenly changed conditions, especially the wind direction. An example of unchanged weather conditions is October 30, 2019, between 05 and 13 UTC. A huge high-pressure system dominates over north central Europe (Fig. 18), resulting in almost constant winds from the south-southeast over the LOWW area.

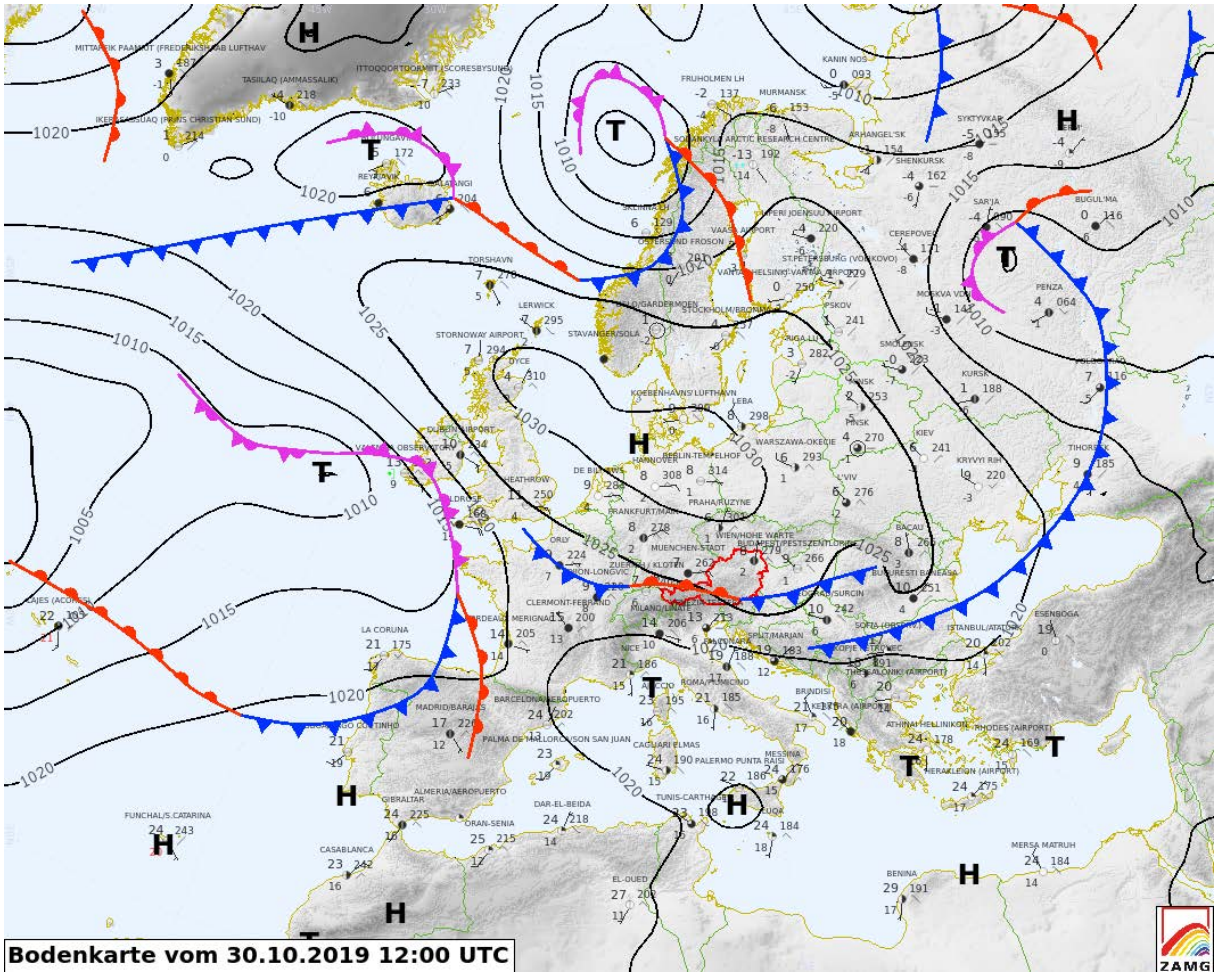


Figure 18: Weather map from 30th October, 2019 12 UTC showing a high pressure system over northern central Europe.

Figure 19 shows a vertical wind profile of the wind direction dd on the same date as an observation with an instance of time between 05z and 13z. The profile indicates an almost constant south-southeast wind along the 6h input and the 2h label domain. The output is shown in figure 20 as a prediction profile. Except for the only small not well-predicted part in the prediction at 12z around 750 m and below, the model is performing overall very well. It is also worth noting that, in general, such small temporary wind direction changes are hard to capture in the prediction.

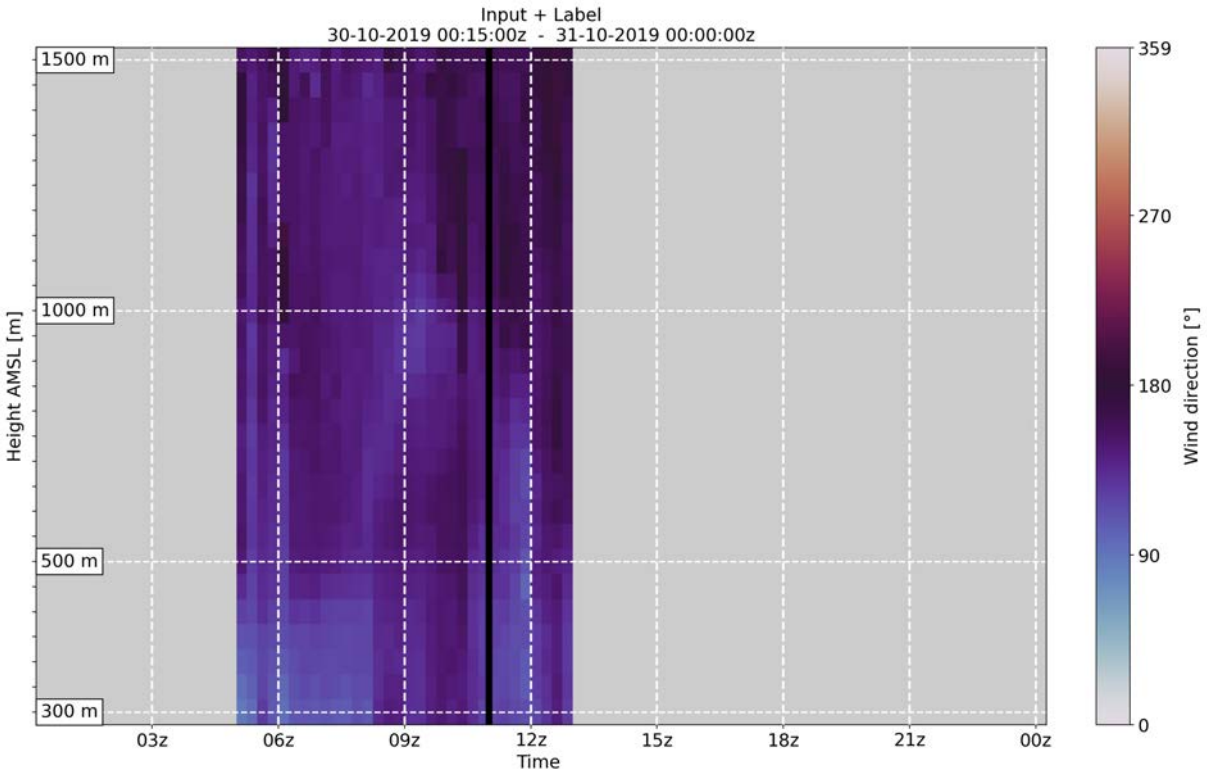


Figure 19: Wind profile of the wind direction observation (Input + Label) from 30th October, 2019 between 05z and 13z.

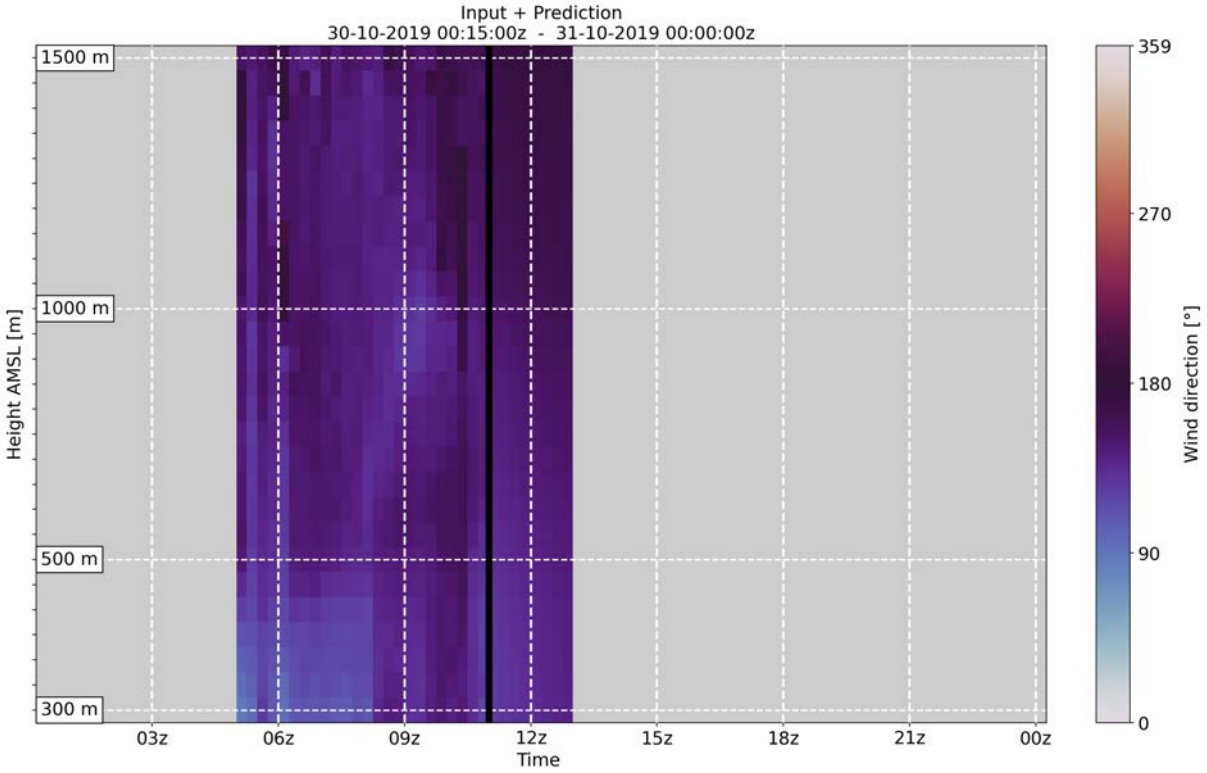


Figure 20: Wind profile of the wind direction observation and prediction (Input + Prediction) from 30th October, 2019 between 05z and 13z.

Taking another example from 21st October, 2019, between 05z and 13z, where LOWW is located in a warm air sector (see fig. 21). Observations in figure 22 show that the relatively strong wind speed between 300 m and 1000 m, with a maximum of 9 m/s, increases slowly from 06z to 13z, while this feature is typical within a warm air sector. This ascending wind profile in the warm air sector is continuous and therefore makes it easy to predict, as shown in figure 23. The wind above 1000 m and below 500 m remains weak, as indicated by the observations.
Further on the same day and time window, the wind direction is also changing clearly from 09z until 13z with the height as shown in figure 24. The model is performing well in the prediction on this ascending part in figure 25, but not after 11z above 1000 m and below 500 m, where the wind data gets sudden changes and becomes chaotic.
The continuous ascending wind direction change left the model no other choice but to predict it in the right way, because such weather conditions like fronts and warm air sectors occur often in the year, and in that case, the model trains well on them. All the sudden small changes below 500 m and above 1000 m, the model had no idea what to predict exactly, because such weak wind shears were unseen before in the input data and therefore they would not be predicted. In this example, the wind direction change would not make a big impact on the aircraft since the wind (see fig. 24) in those parts is quite weak.

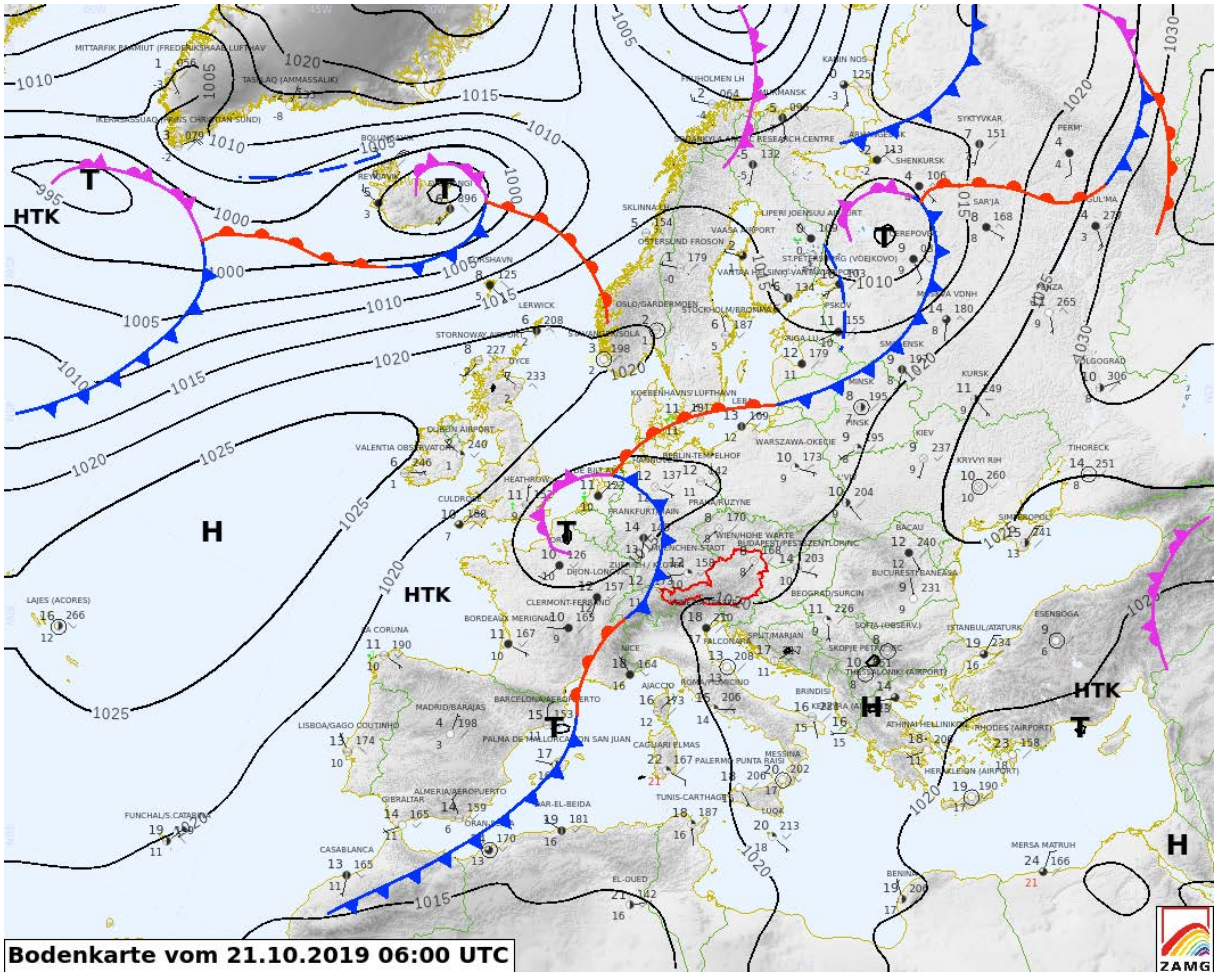


Figure 21: As Fig. 18 but for 21st October, 2019, 06 UTC.

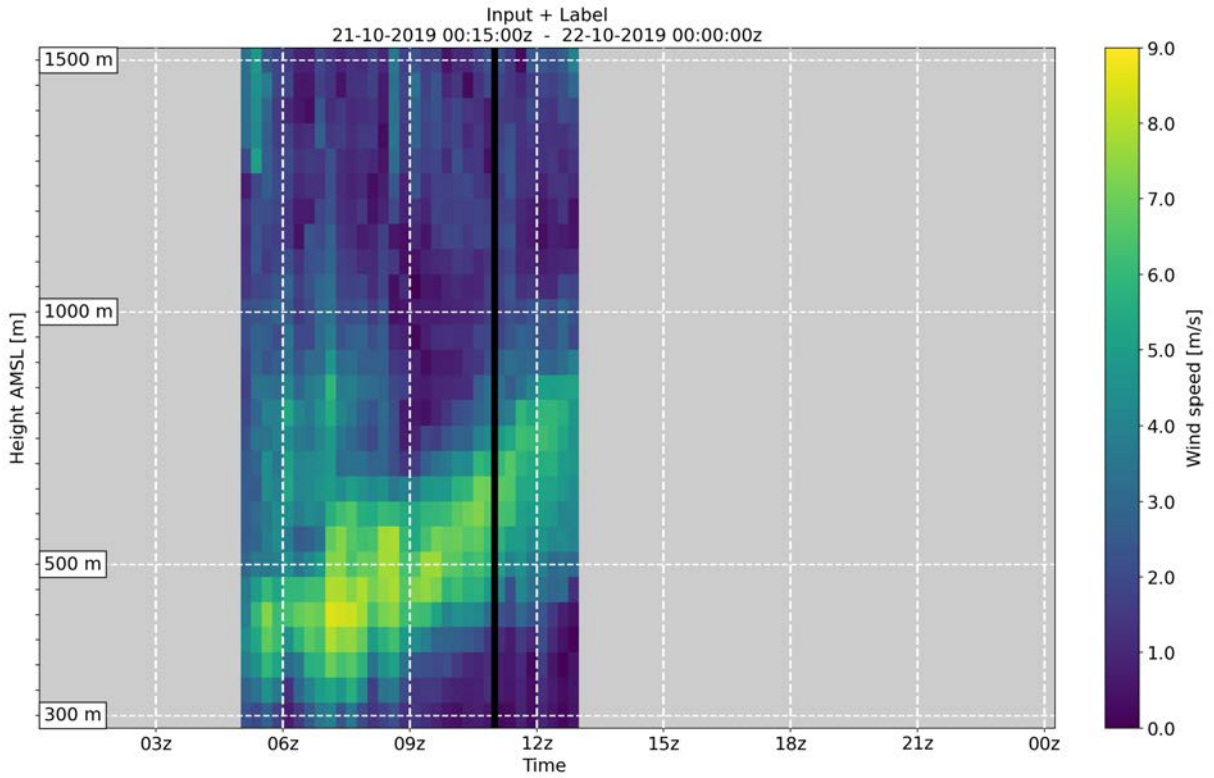


Figure 22: Wind profile of the wind speed observation from 21st October, 2019 between 05z and 13z.

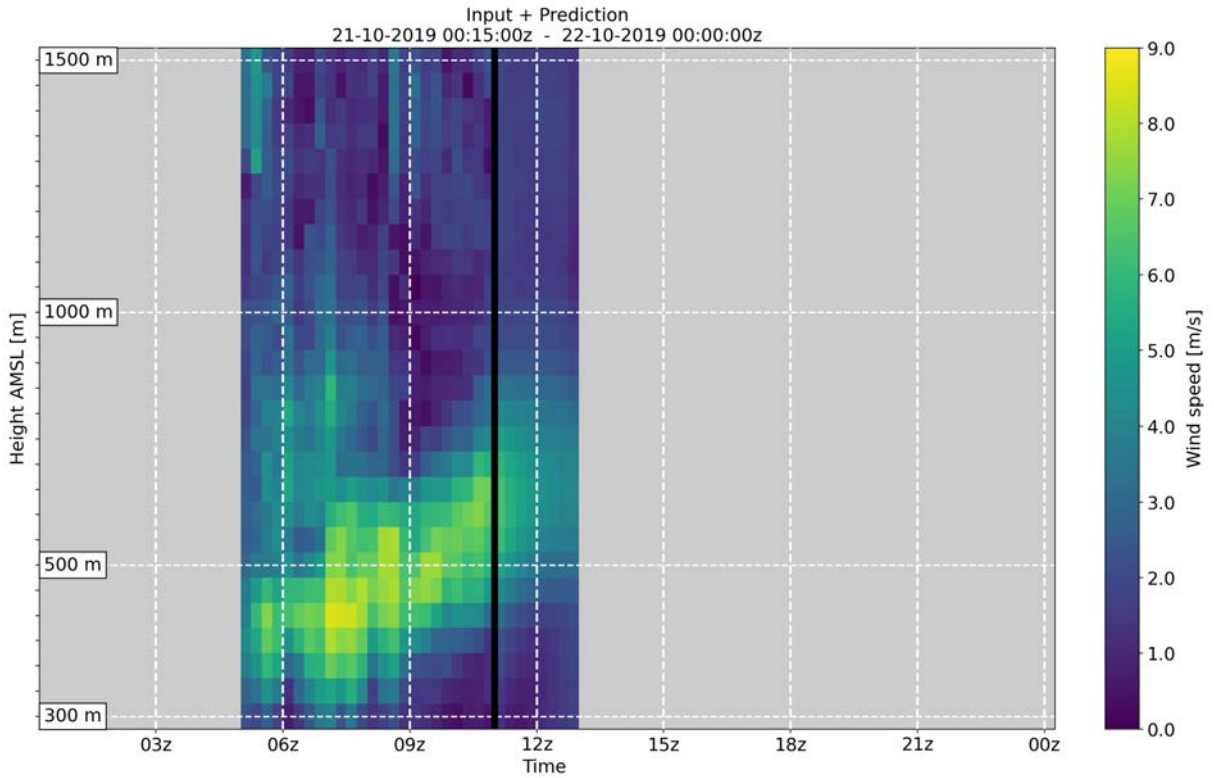


Figure 23: Wind profile of the wind speed observation and prediction from 21st October, 2019 between 05z and 13z.

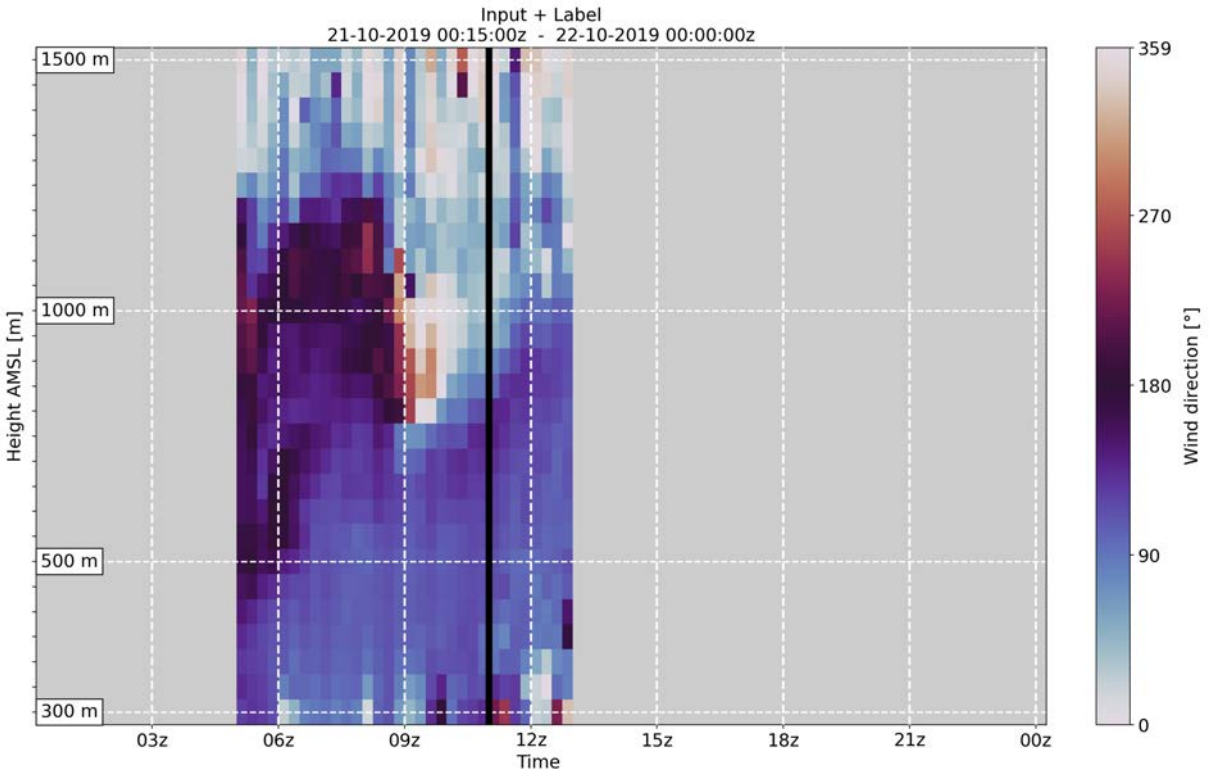


Figure 24: Wind profile of the wind direction observation from 21st October, 2019 between 05z and 13z.

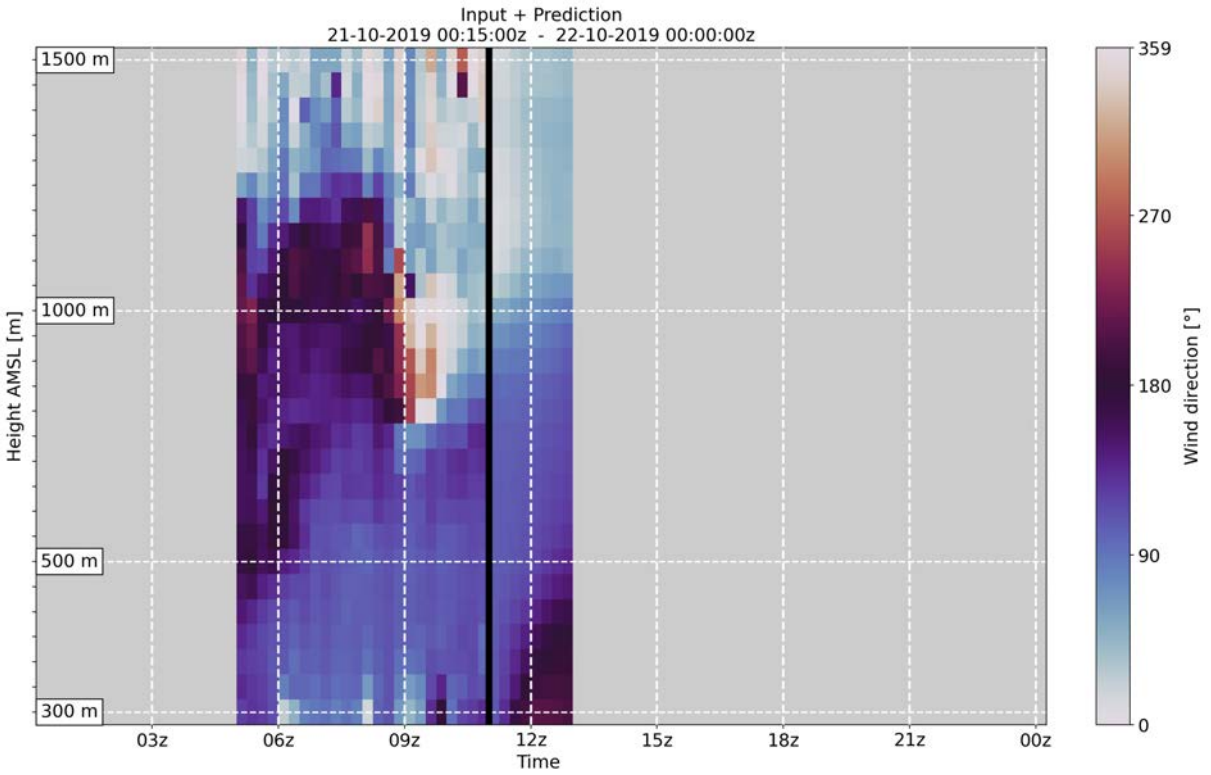


Figure 25: Wind profile of the wind direction observation and prediction from 21st October, 2019 between 05z and 13z.

The following example shows how the prediction will change by shifting the instances forward to the next hour or time steps. The example shows wind direction data from 23rd October, 2019 starting from 05z and then shifting the instances by 4 time steps (i.e. 1 hr). The observations (fig. 26) show a light wind shear below 500 m from south to east-southeast until 12z. After 11z above 1000 m, there is a wind direction change from south to west-southwest. This change is only in the label data (i.e. the reference data for the prediction) visible and not in the input data.
The prediction (fig. 27) couldn't capture the wind shear above 1000 m as it is a sudden wind direction change, which was unseen in the input data. The light wind shear below 500 m is only well predicted between 11z and 12z, as after 12z the wind shear disappears in the observation and therefore the prediction remains showing the same values as this wind shear disappearance was also not seen in the input data. In this case, the model performed very well in the first hour of the prediction only, which also can be useful for the ATM.
Figure 28 showing the next instance of the wind direction observation of the same day by shifting the instance by 4 time steps forward (i.e. 1 hr). Now, in this instance, the first part of the wind shear above 1000 m is visible in the input data, while the light wind shear below 500 m is now subject to completely unseen data.
As expected, in figure 29 the seen wind shear above 1000 m is well predicted, and the model is performing well in this part, while it is performing poorly on the light wind shear below 500 m.

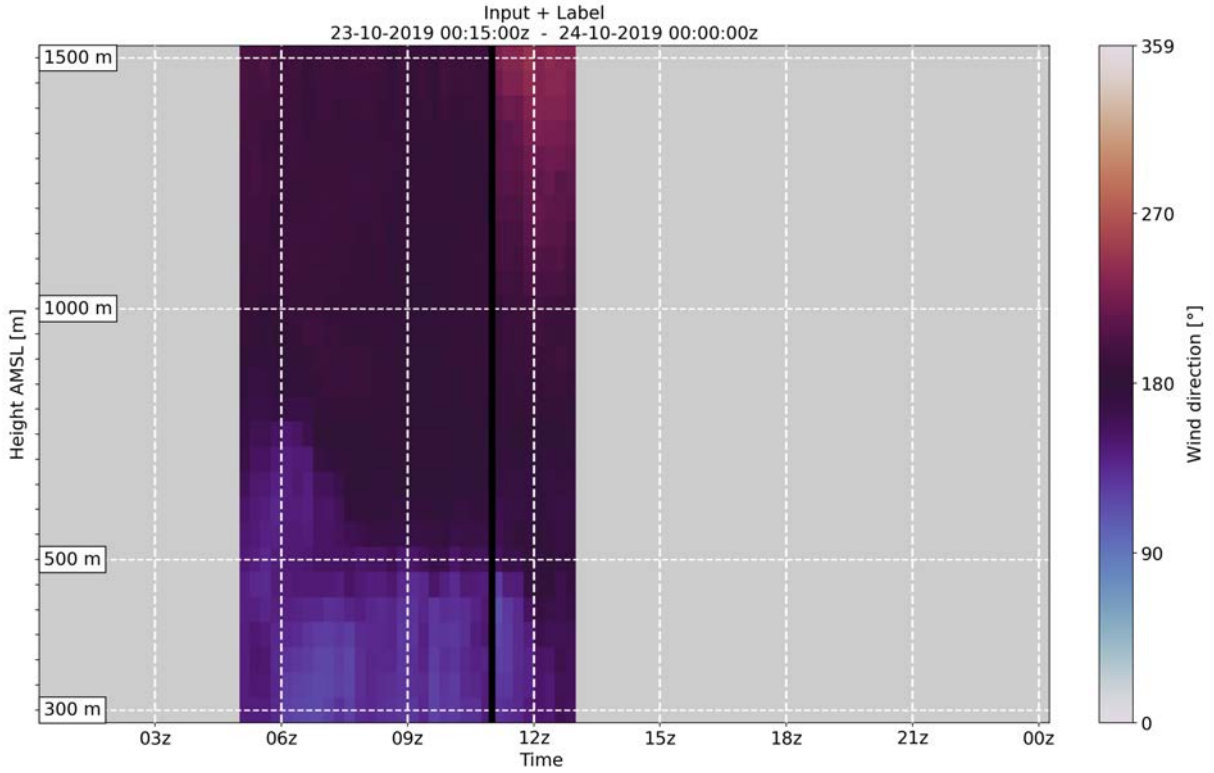


Figure 26: Wind profile of the wind direction observation from 23rd October, 2019 between 05z and 13z.

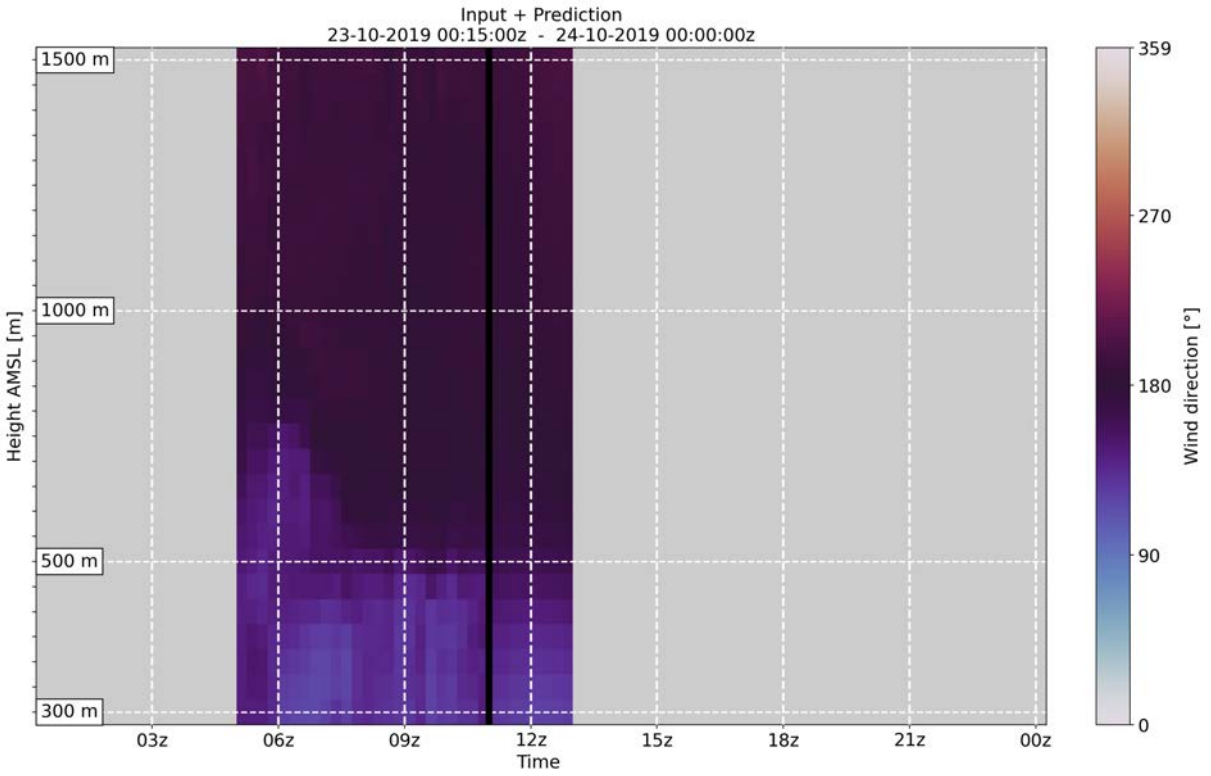


Figure 27: Wind profile of the wind direction observation and prediction from 23rd October, 2019 between 05z and 13z.

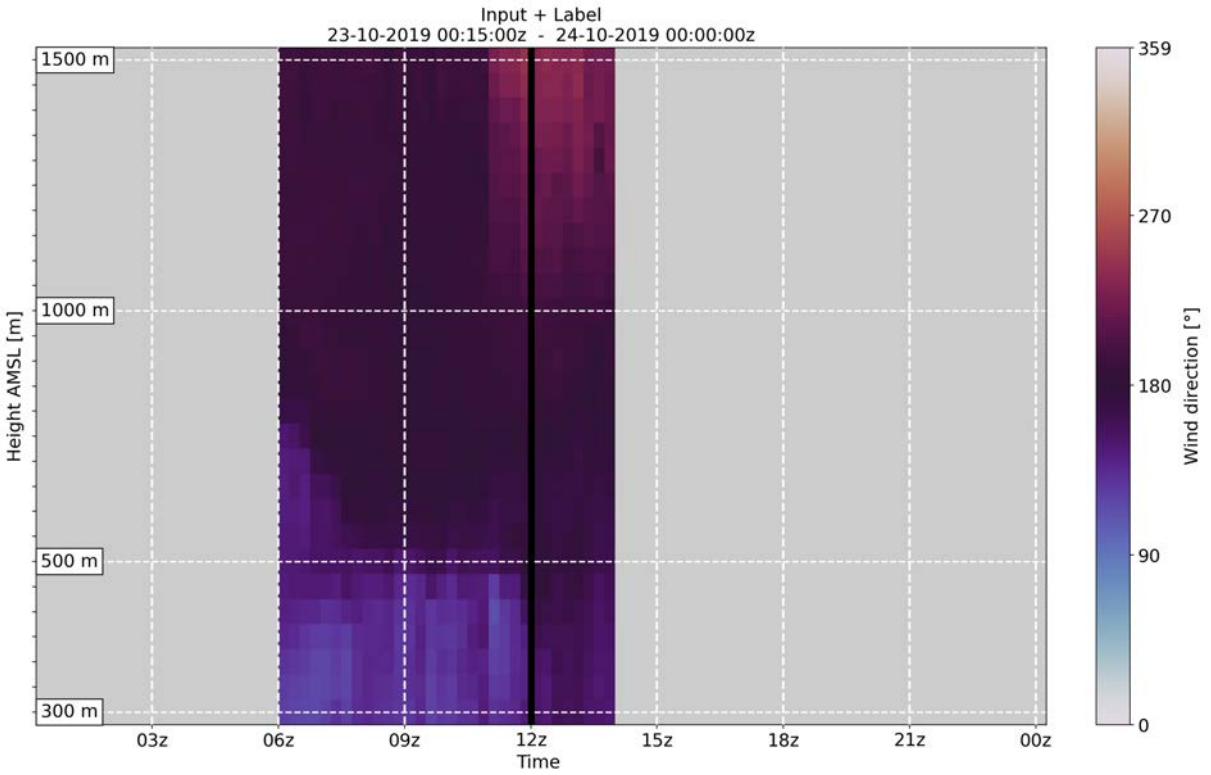


Figure 28: Wind profile of the wind direction observation from 23rd October, 2019 between 06z and 14z.

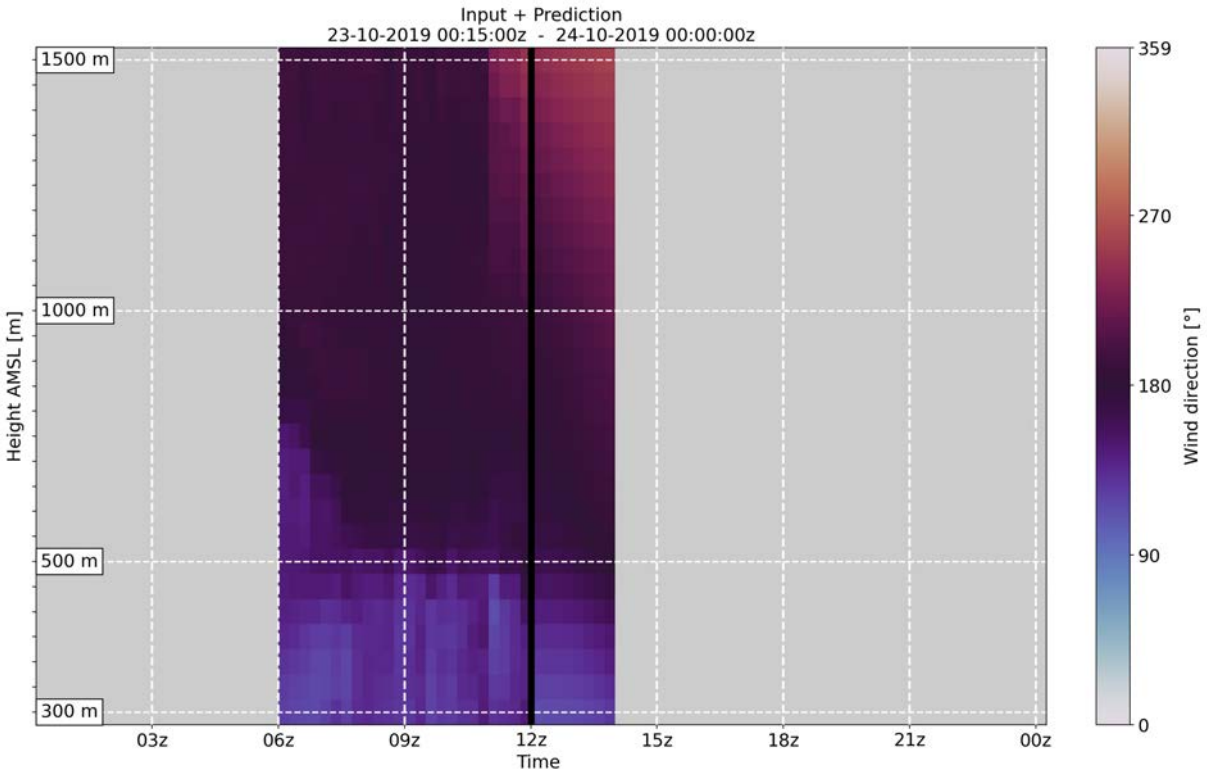


Figure 29: Wind profile of the wind direction observation and prediction from 23rd October, 2019 between 06z and 14z.

4.3 Verification

Verification is important to present statistically how well the model has been trained. It also ensures the model's accuracy and reliability. In this section, the predictions will be compared statistically with the observations to improve model trustworthiness.
Table 1 shows multiple verification metrics of the wind speed ff of the test data:

Table 1: Error metrics of the wind speed ff

Metric	Value
RMSE	2.83 m/s
MAE	2.06 m/s
MAPE	31.25%
R^2	0.765
SS	0.515
IoA	0.935

Comparing the first two metrics tells us much about the model's performance. In general, RMSE has values higher than MAE, due to the penalization of larger errors in RMSE. The difference between RMSE and MAE in table 1 is only 0.76 m/s. This difference suggests that while there are some outliers in the predictions, they are not that extreme. The higher the difference between RMSE and MAE, the more significant outliers are affecting the model.

RMSE can also be compared with the mean of wind speed ff of the observations. Through this comparison, one gets information on the quality of the prediction. The mean wind speed ff of the observations is 10.15 m/s (see table 3 in the appendix); this means that the RMSE of 2.83 is about 28% of the mean. The lower RMSE and MAE relative to the mean wind speed ff, the better the prediction quality. A MAPE of 31.25% indicates that the predictions are off by 31.25%. Higher MAPE values suggest that the model struggles with the variability in wind speed due to the well-known problem of sudden wind changes.
The coefficient of determination (R^2) has a value of 0.765, which indicates that the model explains 76.5% of the variance in the data. The coefficient shows that the model has captured the majority of the patterns in the dataset and is performing well. The remaining 23.5% of the variance remains unexplained.
The last two metrics SS and IoA are represented with values of 0.515 and 0.935 respectively, calculated to expand the comparison of the model's performance. A skill score of 0.515 is larger than 0, indicating that the model is performing 51.49% better than the baseline model. In general, positive values above 0.5 indicate a moderate improvement over the baseline. The IoA of 0.935 compared to the SS and the other metrics tells that the model captures the overall trend of wind speed very well. Despite the high IoA value, there are still some error spikes or inconsistencies.

The next table 2 shows metric results of the wind direction dd.

Table 2: Error metrics of the wind direction dd in °

Metric	**Value**
RMSE	28.56 °
MAE	16.71 °
R^2	0.471
SS	0.272
IoA	0.865

The values of RMSE = 28.56 and MAE = 16.71 (calculated as in eq. 25 and 26) indicate that the predictions of the wind direction deviate from the observations, with an average absolute error of 16.71° and a typical error magnitude of 28.56° when considering large errors. Due to the abrupt changes in wind direction and their large residuals, sudden changes in wind direction have led the model to be unable to predict effectively (see fig. 29). This makes a big difference when it comes to the wind speed. The residuals of the wind speed have a range of 0 to almost 30 m/s, which is much smaller than the residuals of the wind direction. The most noticeable point is that the RMSE is much larger than the MAE, which suggests that large occasional errors are occurring.
There is also an obvious difference between the R^2 for wind speed and wind direction. It accounts for about 47.1 % of the variance in the observations, while leaving 52.9 % of the variance unexplained. This value of R^2 is considerably smaller relative to the R^2 value of the wind speed, indicating that the model is performing worse on wind direction than on wind speed.
Consequently, lower values of SS and IoA for wind direction of 0.272 and 0.865, respectively, were expected. A skill score of 0.272 is almost close to 0, where SS = 0 means that the model performs no better than the baseline. This value of SS answers the question of how the model can deal with large sudden wind changes due to various weather situations, which do not show up in the prediction. The IoA shows only a small difference between wind direction and wind speed of 0.07, which supports the statement that generally the predictions are fairly close to observations.

Further, it is worth looking into the distribution of the wind speed and direction. Figure 30 shows a histogram of the observed wind speeds. The histogram is right-skewed with a peak around 5-10 ms^{-1} and a long right tail, which extends up to 35-40 ms^{-1}. It is also obvious that the extreme wind speed values are less frequent and may indicate an outlier if the values appear at the lowest altitudes. However, the extreme wind speed values shift the 10.15 ms^{-1} mean (see table 3 in the appendix) slightly to the right of the peak.
The distribution of the predicted wind speeds is shown in figure 31. The histogram has a similar shape to that of the observations, but only the frequency peak is lower and the extreme values are extending up to 30-35 ms^{-1}. Hence, the 9.88 ms^{-1} mean of the predictions (see table 4 in the appendix) is a bit closer to the peak. The similarity in shape means that the model has learned the wind speed distribution well. The underestimation of the extreme values and the lower peak in the predictions histogram compared to the observations one explains that the model's accuracy is limited, as discussed in the results section.

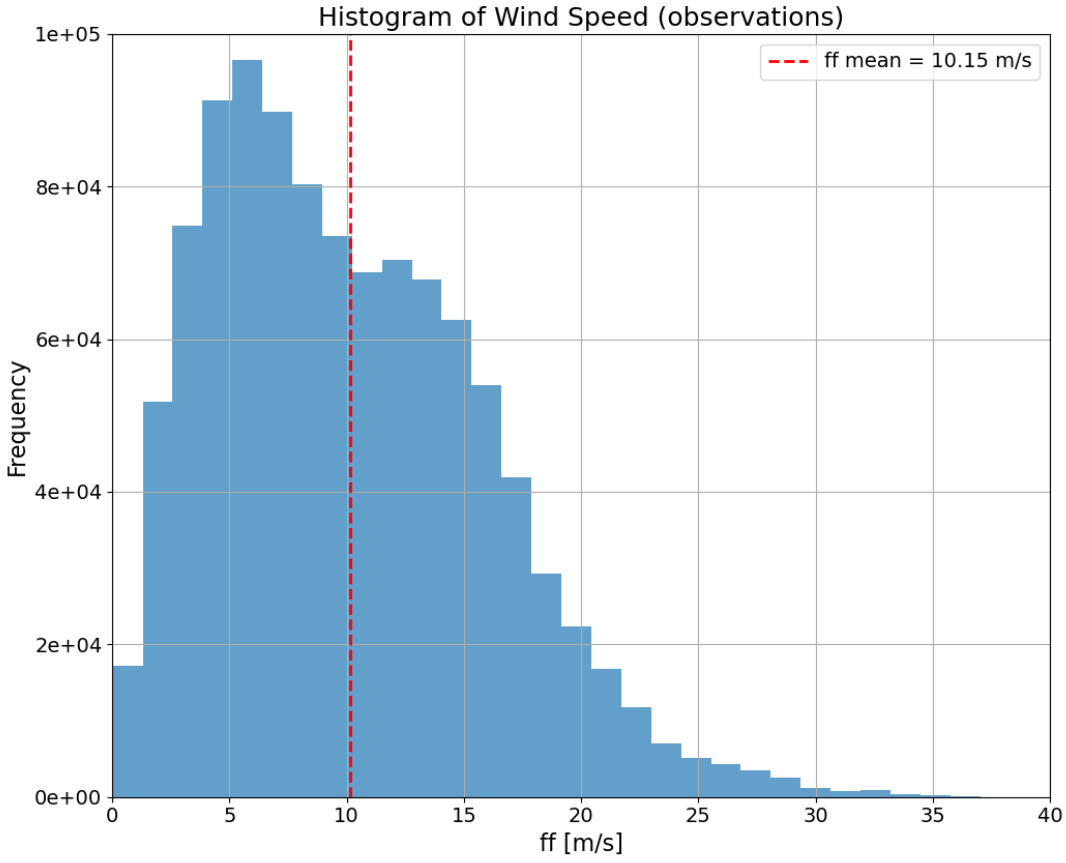


Figure 30: Histogram of the observed wind speed ff in m/s. The x-axis represents the wind speed values and the y-axis the frequency of the wind speed values. The red dashed line represents the mean of the observations dataset.

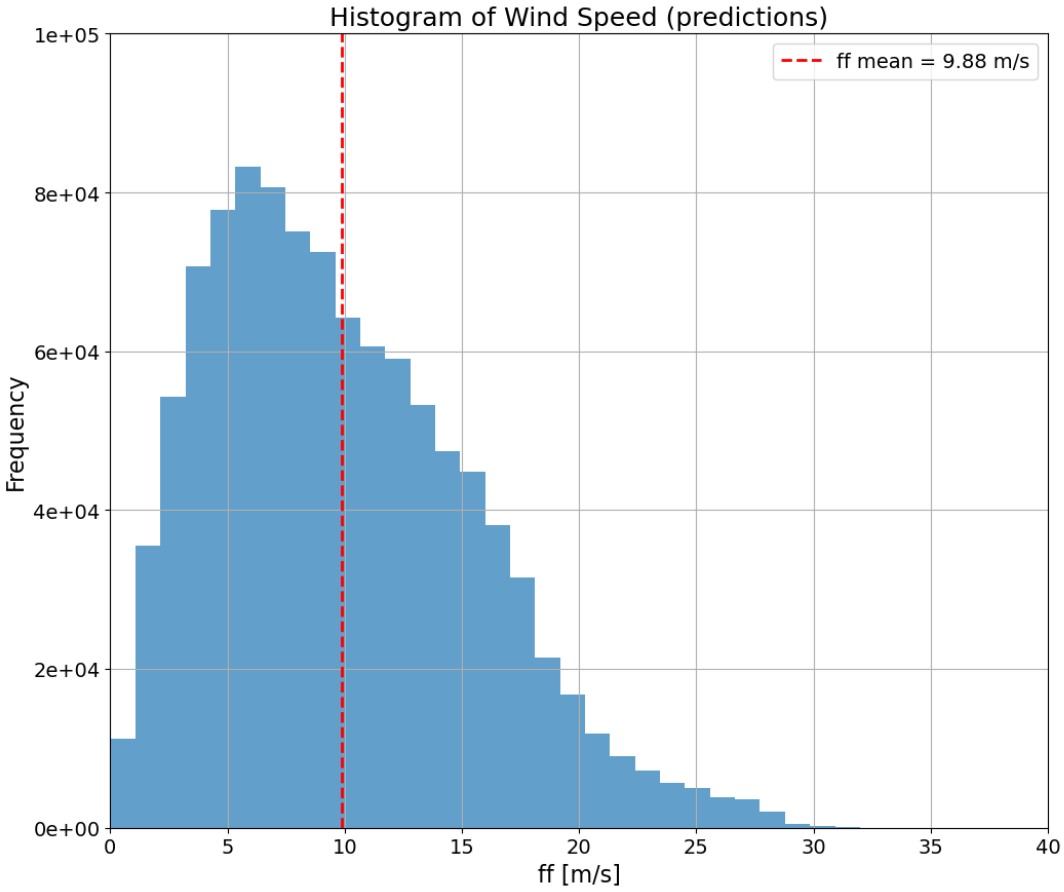


Figure 31: As Fig 30 but for predicted wind speeds

The distribution of the observed wind direction (Fig. 32) has two clear distinct peaks of the wind direction due to the geographic location of LOWW. One peak is around 180° (southerly winds) and the other between 270° and 340° (west-northwesterly winds). While the wind direction of those two peaks is predominant in frequency, the range between 0° and 110° and between 210° and 260° is less frequent. Those two peaks in frequency also represent the average wind direction, upon which the runways were built.
For the wind direction, a circular mean is calculated instead of the standard mean due to the circular nature of the wind direction (0° and 360° are equal). It is also worth observing that the circular mean in the wind rose diagram does not underestimate the northwest dominance. The predicted wind direction distribution (Fig. 33) aligns relatively well, especially as the two predominant peaks in the observations are clearly visible in the wind rose diagram of the predictions. The circular mean in the predictions is slightly shifted to the right compared to the observations due to the very small difference in the frequencies of the northwest dominance.

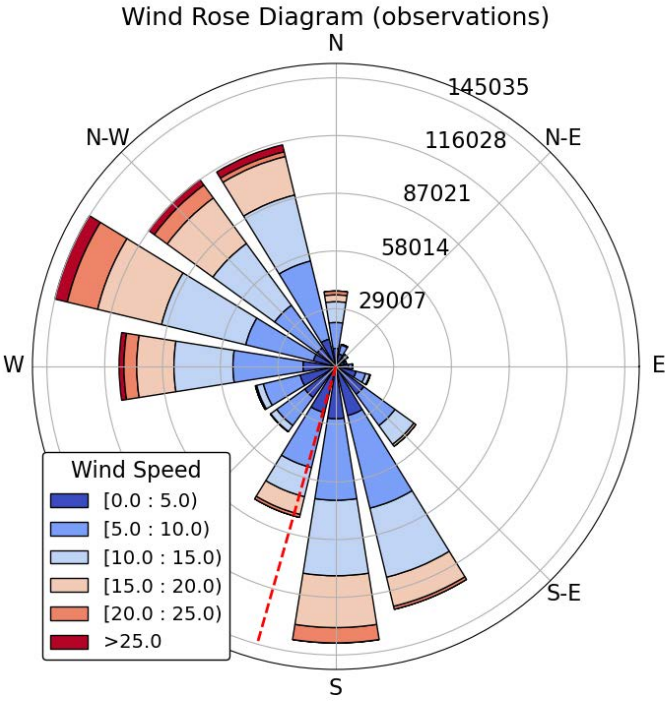


Figure 32: Wind rose diagram illustrating the frequency and intensity of observed wind from different directions. Red dashed line represents the circular mean of the wind direction = 251.6°

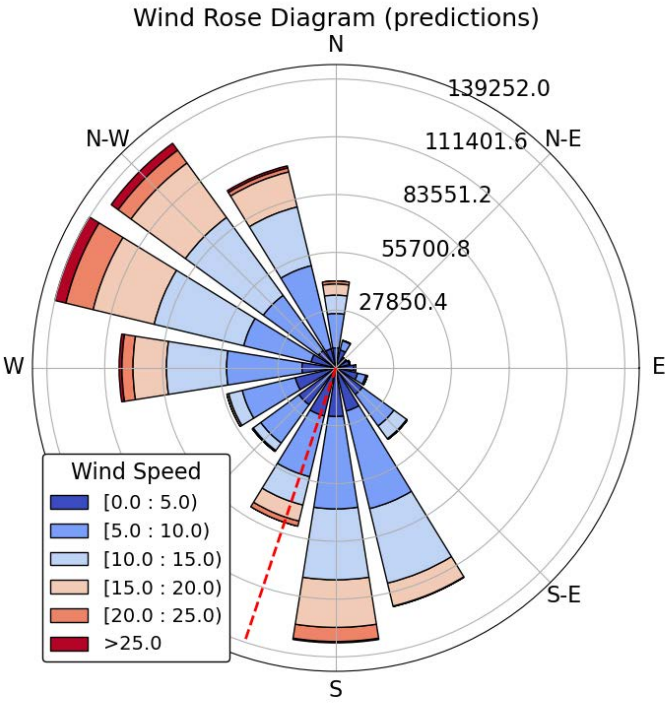


Figure 33: As Fig 32 but for predictions and circular mean of wind direction = 254.2°

In order to spotlight outliers and extreme values in the dataset, a scatter plot is presented in fig. 34, comparing predictions (y-axis) with observations (x-axis). The plot shows data points of a single prediction-observation pair of wind speed ff and the wind direction dd visualized through a color gradient. The density and spread of the points give insight into how closely predictions align with the observations.
In general, there is a positive correlation between observations and predictions. At higher wind speeds, the data points tend to fall below the reference black line (y=x), indicating that the model systematically underestimates higher wind speeds. The intercept of the regression line (1.34) indicates a slight positive bias in the model. This means that, on average, the model predicts wind speeds about 1.34 m/s higher than observations when the wind speed is low. The spread of the points increases as the wind speed increases. This suggests that the model's performance becomes less accurate at higher wind speeds, with great variability in predictions. This also verifies the earlier finding of higher wind speed observations, where the predictions showed a weaker performance.
Wind direction shows no clear pattern of the influence on the model's performance. Interestingly, higher wind speeds are predominantly connected to winds from northwest. This indicates that southeasterly winds are rarely strong compared to northwesterly winds.
Outliers can be identified by applying the Gaussian distribution and defining a threshold of 3σ (red data points in fig. 35), i.e. 99.74%* of the residuals, where σ is simply the standard deviation of the residuals. There is a dense cluster of points at lower wind speeds (x,y < 10), where the model performs reasonably well with many points lying close to the diagonal reference line. The outliers make up approximately 1.32% of the whole data points, which is fairly good.

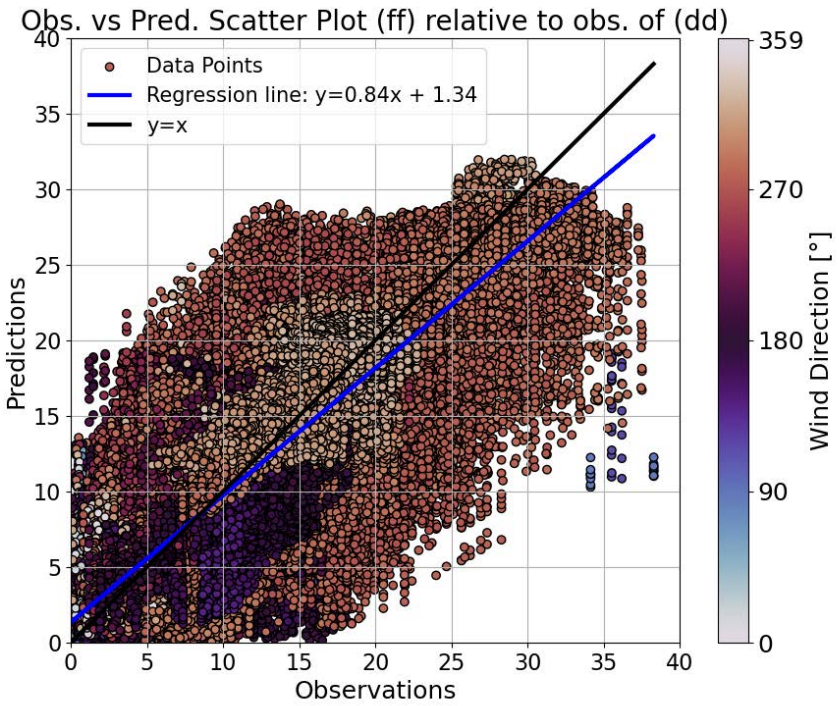


Figure 34: Scatter plot of wind speed ff comparing observations (x-axis) and predictions (y-axis). The color bar represents wind direction in degrees. The black diagonal line is the reference line for the ideal regression. The blue diagonal line represents the actual regression line of the data points.

*refer to the notation and figures of normal distribution of university of notre dame `https://www3.nd.edu/~rwilliam/stats1/x21.pdf` (Last visit: 19/12/24)

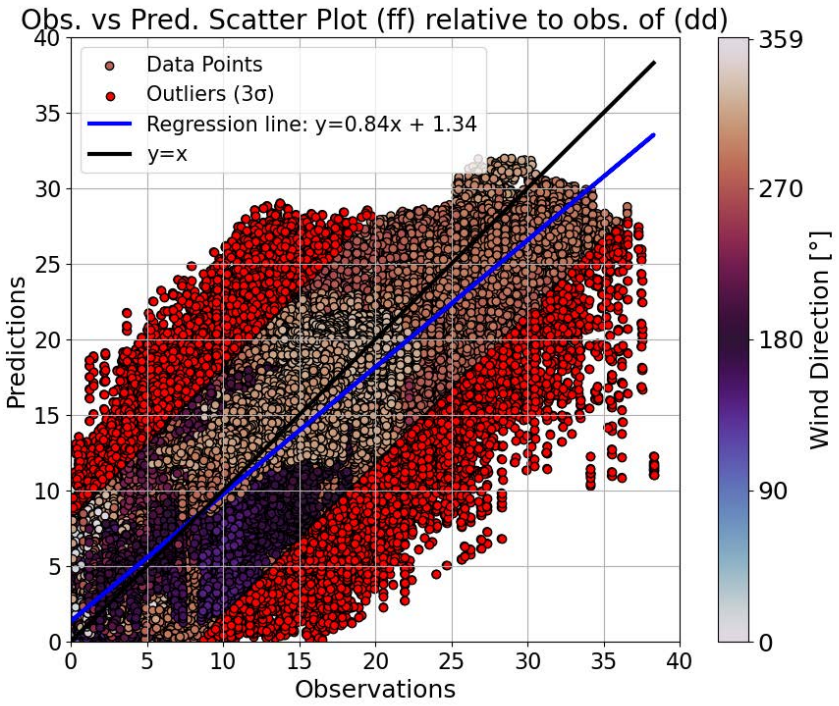


Figure 35: As fig 34 but outliers identified as red data points according to the 3 σ limit.

5. Discussion and Outlook

Predicting the wind using an AI model is challenging, especially when the dataset is limited and has non-continuous time series to train the model. According to deep learning methods, a neural network algorithm needs a continuous dataset without missing values. To deal with those missing values, an appropriate imputation method was applied by using the median filter. The wind speed ff and wind direction dd were converted into u and v components to make the process easier for the calculation of the median filter and later for the training of the model. In order to choose the best window size, several attempts have been applied, and they have shown that a window size smaller than 4 did not effectively fill in missing values, and a window size larger than 4 has smoothed out important data features. Finally, a window size of 4 showed ideal results.
The datasets of this thesis were sequential in time, and the appropriate method among all other methods to use was RNN. As RNN has some disadvantages, like the exploding and vanishing gradients, it was necessary to take the special type of RNN, which is called LSTM.
The choice of input window size of the training model was challenging, due to the limited availability of the data between 5 and 22 UTC. Choosing an input window size of 6 hours (i.e. 24 time steps of 15-min interval) showed the best result, while an input window size longer than 6 hours resulted in higher RMSE values, and an input window size less than 6 hours showed constant RMSE results, but the model would have insufficient information per window or instance to train. The output window size of 2 hours (i.e. 8 time steps) was the best to choose as it fulfilled the aim of the nowcasting and the ATM. The ideal choice of the batch size of 512 has shown good effectiveness and improvement in performance. The choice of the number of epochs can lead to overfitting or underfitting. After several attempts, 110 epochs were the best to select.
In general, the model has shown reasonable accuracy and reliability in predicting stable and standard weather conditions, like standard wind conditions before and after the passage of a weather front or during periods of predominant high pressure systems over central Europe, where wind conditions remain relatively constant. The model's predictions get worse as weather conditions change rapidly, for example, extreme wind shears during the passage of a cold front or within a warm sector, in ascending air masses.
The extreme changes and unexpected weather conditions in the forecast are also reflected by the relatively high MAPE of 31.25% for wind speed. This problem was also verified for the wind direction predictions by a skill score of 0.272, considerably lower than the skill score for wind speed of 0.515, indicating a weaker performance.
Overall, the wind speed dataset shows better results than the wind direction, due to the high variability in wind direction values and their rapid changes; this is also confirmed by the determination coefficient R^2, with a higher value for wind speed of 0.765 compared to wind direction of 0.471. Additionally, the IoA for the wind speed was a bit higher than for the wind direction.
In general, the frequency and intensity of the predictions for both wind speed and wind direction are almost similar to the observations, demonstrated by the histograms and wind rose diagrams; that is because events like fronts, inversions, air mass changes, convergence lines, etc. do not occur on a daily basis.

Using this Nowcasting model in the current state in the ATM could be impractical as it is less effective at providing accurate forecasts during extreme weather conditions and events involving sudden wind changes that are not obtained in the observations dataset. Such events can have severe impacts for approaching aircraft.
The model needs several improvements; for example, supplying it with a larger dataset to ensure

that the model includes more such weather conditions and events during the training. In addition, evaluate the potential expansion of the usual operational period (05-22 UTC) owing to a possible uptick in air traffic volume, while in the midnight hours, air traffic would stay tranquil. All those suggestions for improvements might mark the beginning for further research and development of AI models to make better and more efficient wind predictions for ATM and the entire aviation world in the near future.

References

Arnaud, G., C. Monica, B. I. Nofar, M. Lisa, O. Els, M. Umberto, G. Vida, and S. Aleksander (2023), A new median filter application to deal with large windows of missing data in eye-gaze measurements. In: *CEUR Workshop Proceedings (CEUR-WS.org)*, vol. 3363, pp. 1–17.

Chicco, D., M. Warrens, and G. Jurman (2021), The coefficient of determination r-squared is more informative than smape, mae, mape, mse and rmse in regression analysis evaluation. *PeerJ Computer Science* **7**, e623. doi:10.7717/peerj-cs.623.

Chollet, F. (2021), *Deep Learning with Python*. Manning, 2nd edn. URL: https://www.manning.com/books/deep-learning-with-python-second-edition?a_aid=keras.

Federal Aviation Administration (FAA) (2017), *Instrument Procedures Handbook*, chap. 4. Department of Transportation, faa-h-8083-16b edn., Accessed: 2024-07-01. URL: https://www.faa.gov/sites/faa.gov/files/regulations_policies/handbooks_manuals/aviation/instrument_procedures_handbook/FAA-H-8083-16B.pdf.

Golding, W. L. (2005), Low-level windshear and its impact on airlines. *Journal of Aviation/Aerospace Education & Research* doi:10.15394/jaaer.2005.1530.

Grange, S. (2014), Technical note: Averaging wind speeds and directions. Tech. rep., The University of York.

Hochreiter, S. and J. Schmidhuber (1997), Long short-term memory. *Neural computation* **9**, 1735–1780. doi:https://doi.org/10.1162/neco.1997.9.8.1735.

Hodson, T. O. (2022), Root-mean-square error (rmse) or mean absolute error (mae): when to use them or not. *Geosci. Model Dev.* **15**, 5481–5487. URL: https://doi.org/10.5194/gmd-15-5481-2022.

Huang, L., Y. Zhu, H. Zhai, S. Xue, T. Zhu, Y. Shao, Z. Liu, C. Emery, G. Yarwood, Y. Wang, J. Fu, K. Zhang, and L. Li (2021), Recommendations on benchmarks for numerical air quality model applications in china – part 1: $Pm_{2.5}$ and chemical species. *Atmospheric Chemistry and Physics* **21**(4), 2725–2743. URL: https://acp.copernicus.org/articles/21/2725/2021/, doi:10.5194/acp-21-2725-2021.

Jiménez, P. A. and J. Dudhia (2013), On the ability of the wrf model to reproduce the surface wind direction over complex terrain. *Journal of Applied Meteorology and Climatology* **52**(7), 1610 – 1617. URL: https://journals.ametsoc.org/view/journals/apme/52/7/jamc-d-12-0266.1.xml, doi:10.1175/JAMC-D-12-0266.1.

Kim, S. and H. Kim (2016), A new metric of absolute percentage error for intermittent demand forecasts. *International Journal of Forecasting* **32**(3), 669–679. URL: https://www.sciencedirect.com/science/article/pii/S0169207016000121, doi:https://doi.org/10.1016/j.ijforecast.2015.12.003.

Klimenko, V. and J. Krozel (2009), *Impact Analysis of Clear Air Turbulence Hazards*, p. 6067. Aerospace Research Central. URL: https://arc.aiaa.org/doi/abs/10.2514/6.2009-6067, arXiv:https://arc.aiaa.org/doi/pdf/10.2514/6.2009-6067, doi:10.2514/6.2009-6067.

Liu, B., J. Guo, W. Gong, L. Shi, Y. Zhang, and Y. Ma (2020), Characteristics and performance of wind profiles as observed by the radar wind profiler network of china. *Atmospheric Measurement*

Techniques **13**(8), 4589–4600. URL: https://amt.copernicus.org/articles/13/4589/2020/, doi:10.5194/amt-13-4589-2020.

Mahesh, B. (2019), Machine learning algorithms -a review. *International Journal of Science and Research (IJSR)* **9**. doi:10.21275/ART20203995.

Ribbens, W. (2003), Aircraft instruments. *ScienceDirect* pp. 337–364. URL: https://www.sciencedirect.com/science/article/pii/B0122274105009169, doi:10.1016/B0-12-227410-5/00916-9.

Sarker, I. H. (2021), Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science* **2**, 420. URL: https://doi.org/10.1007/s42979-021-00815-1.

Sazli, M. (2006), A brief review of feed-forward neural networks. *Communications Faculty Of Science University of Ankara* **50**, 11–17. URL: https://doi.org/10.1501/commua1-2_0000000026.

Schmidt, R. M. (2019), Recurrent neural networks (rnns): A gentle introduction and overview. URL: https://arxiv.org/abs/1912.05911, arXiv:1912.05911.

Sun, J., H. Vû, J. Ellerbroek, and J. Hoekstra (2017), Ground-based wind field construction from mode-s and ads-b data with a novel gas particle model. In: *Seventh SESAR Innovation Days Conference*.

Sun, J., H. Vû, J. Ellerbroek, and J. Hoekstra (2018), Weather field reconstruction using aircraft surveillance data and a novel meteo-particle model. *PLoS ONE* doi:10.1371/journal.pone.0205029.

Wheatcroft, E. (2019), Interpreting the skill score form of forecast performance metrics. *International Journal of Forecasting* **35**(2), 573–579. URL: https://www.sciencedirect.com/science/article/pii/S0169207019300093, doi:https://doi.org/10.1016/j.ijforecast.2018.11.010.

Yin, W., K. Kann, M. Yu, and H. Schütze (2017), Comparative study of cnn and rnn for natural language processing. URL: https://arxiv.org/abs/1702.01923, arXiv:1702.01923.

List of Figures

List of Tables

Acknowledgement

I would like to express my gratitude to my supervisor, Ass.Prof.Dr. Manfred Dorninger, for his invaluable guidance and support throughout this thesis.
A special thanks to Dr. Lukas Strauß and Dr. Johannes Sachsperger, whose initial idea laid the foundation for this work and who provided essential support, particularly with wind profile codes. I am also grateful to Austrocontrol GmbH for supplying the data and supporting the development of the thesis concept.
I sincerely appreciate the help of my university colleagues. Markus Rosenberger was an immense source of support, offering both technical assistance and code corrections. Likewise, Martin Skopalik contributed greatly through insightful discussions and coding support. I would also like to thank Dipl.-Ing. Dr.techn. Lukas Miklautz for his expert advice on the choice of methods and for providing valuable materials.
Beyond the academic sphere, I am deeply grateful to my partner, Jana Schönach, for her unwavering motivation and mental support. Finally, I extend my heartfelt thanks to my family for their constant encouragement and belief in me.
This thesis would not have been possible without the contributions of all these individuals, and I am truly thankful for their support.

Table 3: Metrics and parameter for the observations of wind speed ff

Parameter	Value
mean	10.15 m/s
median	9.32 m/s
standard deviation	5.84 m/s
variance	34.13 $(m/s)^2$
minimum	0.02 m/s
maximum	38.29 m/s
quartiles	[5.47, 9.31, 14.12] m/s
rms	11.71 m/s

Table 4: Metrics and parameter for the predictions of wind speed ff

Parameter	Value
mean	9.88 m/s
median	9.03 m/s
standard deviation	5.58 m/s
variance	31.21 $(m/s)^2$
minimum	0.005 m/s
maximum	31.97 m/s
quartiles	[5.49, 9.03, 13.58] m/s
rms	11.35 m/s

Table 5: Metrics and parameter for the observations of wind direction dd in [°]

Parameter	Value
circular mean	251.6
median	253.2
circular standard deviation	83.1
minimum	0.003
maximum	359.984
quartiles	[172.8, 253.2, 302.7]

Table 6: Metrics and parameter for the predictions of wind direction dd in [°]

Parameter	Value
circular mean	254.2
median	252.0
circular standard deviation	81.4
minimum	0.0001
maximum	359.999
quartiles	[174.0, 252.0, 303.5]