Nowcasting of surface wind speed using probabilistic, explainable Deep Learning

---

Francesco Zanetta

April 6, 2021

Master Thesis

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Department of Environmental Systems Sciences
MSc Environmental Sciences
Atmospheric Dynamics group

Nowcasting of surface wind speed using probabilistic, explainable Deep Learning

Francesco Zanetta
15-422-702

1. Supervisor Dr. Daniele Nerini
Division of Development of Forecasting
MeteoSwiss

2. Supervisor Dr. Michael Sprenger
Department of Environmental Systems Sciences
ETH Zurich

April 6, 2021
Master Thesis

Francesco Zanetta
Nowcasting of surface wind speed using probabilistic, explainable Deep Learning
Master Thesis, April 6, 2021
Supervisors: Dr. Daniele Nerini and Dr. Michael Sprenger

ETH Zurich
Atmospheric Dynamics group
MSc Environmental Sciences
Department of Environmental Systems Sciences
Universitätstrasse 16
8006 Zürich

Abstract

Surface wind is an extremely difficult parameter to predict, particularly in the complex topography of the Alps. Due to several important processes happening at sub-kilometer scale, even high resolution Numerical Weather Prediction models such as COSMO-1 still present substantial biases. To address this, a wide range of statistical post-processing methods are used. Recently, methods based on Deep Learning have emerged as a new solution and are now actively developed at many weather services, including MeteoSwiss. At the same time, efforts are made to obtain accurate representations of surface wind speed up to a few hours ahead by integrating all available information in real-time, an approach known as nowcasting.

With the aim of seamlessly combining nowcasting and post-processing approaches for surface wind speed predictions, we developed a Deep Learning probabilistic post-processing model that is also able to integrate real time observations, and developed a new metric, the Similarity Index, for this purpose. The Similarity Index is a way to estimate the correlation of surface wind speed between two locations, based on their position and geomorphological setting, and can be used to chose the best available observation to be used at any point in space at any given time, and weigh that observation in a way that mimics geostatistical interpolation methods. The proposed methodology yields improved forecasts of wind speed where both systematic and random errors are reduced, thanks to the post-processing and nowcasting components respectively. In a second phase, we implemented a state-of-the-art explainability framework for machine learning, SHAP, and presented how it can be used to get insights into the model and build trust in the results.

Riassunto in italiano

I venti di superficie sono un parametro estremamente difficile da prevedere, in particolare nella complessa topografia delle Alpi. A causa di diversi importanti processi che avvengono su scale inferiori al chilometro, anche modelli numerici di previsione ad alta risoluzione come COSMO-1 presentano ancora errori significativi. Per ovviare a questo problema, esistono svariati metodi di correzione statistica (post-processing). Recentemente, metodi basati sul Deep Learning sono emersi come una nuova soluzione, e sono attualmente in sviluppo presso molti servizi meteorologici, incluso MeteoSvizzera. Al contempo vengono fatti sforzi per ottenere rappresentazioni accurate dei venti di superficie fino ad un orizzonte temporale di poche ore, integrando tutte le informazioni disponibili in tempo reale, con un approccio noto come nowcasting.

Nell'intento di combinare nowcasting e post-processing per le previsioni dei venti di superficie, abbiamo sviluppato un modello di post-processing con Deep Learning probabilistico che è anche in grado di integrare osservazioni in tempo reale, e a tale scopo abbiamo sviluppato una nuova metrica, il Similarity Index. Il Similarity Index è un modo di stimare la correlazione del vento di superficie tra due località, basandosi sulla loro posizione e il tipo di topografia nella quale si trovano, e può essere usato per scegliere la miglior osservazione disponibile per una previsione a un qualsiasi luogo e in qualsiasi momento, dando poi un peso a detta osservazione in un modo analogo ai metodi di interpolazione geostatistica. La metodologia proposta risulta in migliori previsioni dei venti di superficie, per i quali sia gli errori sistematici che quelli aleatori sono ridotti, grazie rispettivamente alle componenti di post-processing e nowcasting. In una seconda fase, abbiamo implementato un sistema all'avanguardia per spiegare i modelli di machine learning, SHAP, e presentato come questo possa essere usato per ottenere informazioni sul modello e aumentare la nostra fiducia nei suoi risultati.

Acknowledgements

My greatest gratitude goes to my supervisor at MeteoSwiss Dr. Daniele Nerini for making this Master Thesis happen in the first place and for his constant support through every phase of my work. I also thank my ETH supervisor Dr. Michael Sprenger for his precious feedback. I thank the entire APPP team for their comments during our Fridays discussions, and for giving me the chance to improve my communication of scientific contents. Many other collaborators at MeteoSwiss were happy to help when I asked them to, so I am thankful to have worked with such a great organization. A special thanks to my family, particularly my mother Elena with whom I spent these difficult times caused by the global pandemic, for keeping me company and supporting me.

Contents

List of Figures

List of Tables

1 Background

1.1 Motivation

The accurate representation of surface wind is valuable for a wide range of applications. Wind extremes are among the most destructive natural hazards, and the prediction of surface winds is a tool for civil protection, as it is used to develop a reliable early warning system, which is an essential measure of risk reduction (Sättele et al., 2016). With the ongoing transition to renewable energy, decision making in the energy industry is also dependent on skillful weather forecasts, notably for grid load balancing and power trading (Usaola et al., 2004). Wind power generation in particular is strongly affected by the high spatial and temporal variability of surface wind. Other specific applications include runway operations in aviation (Kuikka, 2009), planning in the transport sector or estimates of snow accumulation for avalanche services (Lehning and Fierz, 2008). In all these cases, the availability of accurate wind surface analyses and forecasts is crucial even within a very short period of time. This motivates the development of nowcasting systems, whose goal is to produce high spatial and temporal resolution analyses and forecasts of weather developments for present time (analysis) and the next few minutes up to typically a maximum of six hours ahead. This is done by combining all available information (measurements and the latest model forecasts) in real time, with a special attention on computational efficiency due to operational time constraints. Compared to other meteorological parameters such as temperature, cloud cover or precipitation, nowcasting surface wind is a more challenging task, due to its high variability combined with the lack of spatially continuous observations, and there is currently no established methodology for this task. Finally, quantitative information on the uncertainty of a prediction is becoming increasingly valuable. It promotes informed decision-making and allows the user to choose its own relevant probability threshold (Fundel et al., 2019), it facilitates and even improves decision-making (Joslyn and LeClerc, 2013) and can increase the thrust

in forecasts (LeClerc and Joslyn, 2015). For these reasons, there's a growing need to develop probabilistic forecasts.

1.2 Limitations of Numerical Weather Prediction (NWP)

Wind forecasts are generally produced by numerical weather prediction (NWP) models, such as COSMO-1. NWP models solve Navier Stokes and thermodynamic equations on a discrete grid, thus producing physically consistent forecasts. This is computationally feasible because of several simplifying assumptions, but it results in forecast errors. Specifically, model structural errors include the missing or poor representation of sub-grid processes (due to a too coarse grid) and inaccuracies with the numerical scheme (Nicolis et al., 2009). Furthermore, NWP models suffer from high sensitivity to initial conditions due to the chaotic nature of the atmosphere (Vannitsem, 2017), and boundary conditions can also induce significant errors (Nicolis, 2007). In the last decades, NWP models have improved considerably under several key aspects. Better physical parameterisations have reduced the errors resulting from model simplification, the development of complex data assimilation systems improved model initialization and the ongoing adoption of ensemble forecasts allows to estimate the forecast uncertainty (Bauer et al., 2015). Despite all these progress, NWP model forecasts still display substantial biases. In particular, shortcomings in horizontal resolution and physical parameterization make it impossible for operational NWP models to resolve complex processes that occur on fine spatial scales and characterize surface wind fields. These are related to a combination of sub-grid local flow patterns resulting from crest speedup, flow channelling, flow blocking, updraft and downdraft zones, or flow separation downwind of a ridge crest (Lewis et al., 2008). In other words, surface wind fields are strongly influenced by both topography and land cover, meaning that in a complex topography such as the one of the alpine area NWP models are particularly prone to errors. Another important drawback of NWP, particularly in the nowcasting range, is the latency between the availability of the model output and the initialization time, which results from the computational delay and the time lag between each update cycle of the model (for instance, the COSMO-1

model currently updates every 3 hours and takes about an hour to compute). This implies that any forecast is inevitably based on old information, and this is critical in rapidly changing conditions.

1.3 Post-processing techniques

The deficiencies of NWP models described in the previous section induce two kinds of error: systematic and random. Both errors require post-processing in order to improve the forecast quality, respectively by correcting systematic biases and adapt the dispersion in the case of ensemble forecasts. A wide range of statistical techniques is available for this purpose (D. S. Wilks, 2011; Vannitsem, D. Wilks, et al., 2018). The vast majority of approaches consists in statistically relating the NWP model output and other additional data, such as topographic descriptors or seasonality, to observations. The first applications of these techniques were based on simple linear regression, e.g. the well known Model Output Statistics (MOS). Nowadays there's a bloom of post-processing techniques, particularly for probabilistic forecasts, as illustrated by Vannitsem, Bremnes, et al. (2020) in a comprehensive review. Most of the new developments are based on Machine Learning (ML) techniques, and Artificial Neural Networks (ANNs) are proving to be suitable for post-processing. Rasp and Lerch (2018) found that ANNs can significantly outperform traditional post-processing techniques, while being less computationally demanding. The authors highlight that ANN can better incorporate non-linear relationships in a data-driven fashion, and thanks to their flexibility are more suited to handle the increasing amounts of model and observation data. Promising results were also obtained by Weingart (2018) using a similar technique. Cervone et al. (2017) also illustrate how these approaches can be efficiently implemented on massively parallel supercomputers. ANNs have also been combined with other statistical techniques such as Bernstein polynomials (Bremnes, 2020). More sophisticated ANNs, such as Convolutional Neural Networks (CNNs) allow a better use of spatial information. Grönquist et al. (2020) used CNNs to improve forecasts of global weather. Schär (2019), Höhlein et al. (2020), and Veldkamp et al. (2020) used CNNs for spatial downscaling of surface wind field. A process-specific application was proposed by Chapman et al. (2019), with the goal of improving the prediction

of atmospheric rivers[1]. Dai (2020) implemented Generative Adversarial Network based on CNNs to produce physically realistic post-processed forecasts of cloud cover. It is important to understand that there is no single best solution for post-processing of NWP forecasts. Depending on the application, different approaches may prove more suited. For example, Höhlein et al. (2020) and Dai (2020) used CNNs by interpreting post-processing as an *image-to-image* problem because the target was a spatially continuous field, but this is often not the case, particularly for surface wind. In addition, an important distinction must be made between local and global approaches. Local approaches are used to post-process a forecast at a single location, thus a *site-specific* model is used. On the other hand, global approaches aim to be able to make prediction at any point in space using a single model with generalizing capabilities.

1.4 Surface wind nowcasting

While the post-processing of surface wind forecasts is done both with a local and global approach, research in surface wind nowcasting has focused on the former. This was driven primarily by the domain of wind power forecasting, where the problem is interpreted as time-series prediction. Jung and Broadwater (2014) presents an overview of the existing research in short-term wind forecasting with a local approach. In recent years, several techniques in the family of artificial intelligence are emerging in short-range weather forecasting, with promising results (Papazek et al., 2020). Compared to local approaches, there are no established methods for surface wind nowcasting at any point in space. The currently operational wind nowcasting system at MeteoSwiss is a global deterministic model that incorporates a three-steps algorithm combining an ANNs, statistical regression and a spatial interpolation scheme (Buzzi et al., 2019). It is an example of how methodologies from post-processing and local nowcasting can be combined.

[1] An atmospheric river is a narrow corridor or filament of concentrated moisture in the atmosphere

1.5 Aim and outline

ANNs for wind post-processing and nowcasting are actively developed at many weather services, and this results in incremental improvements while also raising a wide range of interesting research questions. Therefore, on one hand there is the importance to follow up on recent efforts, on the other the liberty to investigate different aspects of these new approaches. Drawing from these two aspects, this work aims to: develop a ANN-based post-processing tool that is also suitable for nowcasting, capable of correcting the climatological bias of a model while also integrating real-time information to reduce random errors; implement state of the art explainability techniques, in order to evaluate and interpret our models based on prior knowledge. The rest of this report will be structured as follows: Chapter 2 will give a brief introduction to machine learning, with a focus on concepts specific to deep learning, in order to facilitate readers that are not familiar with these techniques; Chapter 3 presents datasets and methodologies used in this project; in Chapter 4 we will discuss our results. We will evaluate the performance of our models, both in a general way and with a focus on nowcasting, then present several examples of how model explainability techniques may be used. Finally, in Chapter 5 we will draw our final conclusions and present a brief outlook on further developments in the future.

2 Key concepts of Deep Learning

This chapter presents a brief introduction to Deep Learning, including an overview of some key concepts of Machine Learning used in this study. Machine Learning refers to a wide range of statistical methods that use computer algorithms to improve automatically through experience, and Deep Learning is one of its branches, where ANNs are used. This section does not aim to give a complete and formal introduction to Deep Learning, but rather a simplified summary for non-practitioners. For a complete review, see e.g. Goodfellow et al. (2016) or Chollet (2017).

2.1 Optimization

The central problem of most machine learning models is to *meaningfully transform data*. That is, transforming the input data into representations (different ways to look at data) that are meaningful with respect to the expected output. A model finds the best way to transform data by minimizing a cost function, often called loss and noted $\mathcal{L}$, which determines how well the model is performing with respect to the true solution. This cost function is chosen according to the task at hand, e.g. in the case of regression the Mean Absolute Error or the Mean Squared Error are often used. Since it is not always possible to find an analytical solution to minimize $\mathcal{L}$, the key idea of machine learning is to find the best approximate solution by using a recursive optimization algorithm: a model is exposed to known examples of input and expected output, and after each exposure the model parameters are updated such that the transformation applied to the input data results in a representation that is closer to the expected output. An important intuition about how the *learning* occurs, is that there's no creativity in finding the correct transformations: during optimization the algorithm is merely searching through a pre-defined space of possibilities called the *hypothesis space* of the model, using the loss as a guidance

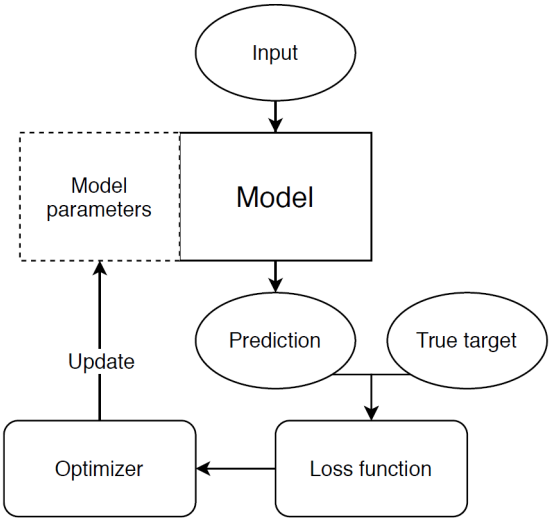

Fig. 2.1.: A generic framework used for most machine learning applications. The automatic improvement of the model occurs through the optimizer, which updates the model parameters after each exposure to input and true target (a training step), based on a feedback signal obtained by the loss function $\mathcal{L}$.

signal. A generic framework that applies to most machine learning applications is represented in figure 2.1.

In the case of deep learning, the *transformation* of the input data occurs between successive layers of representation. The term "deep" stands for the idea of having many layers, which form the core structure of an ANN. What determines the transformation applied by each layer is the layer's parameters or weights, and the learning for an ANN means essentially finding the correct combination of weights for each layer. To perform the adjustments, ANNs typically adopt gradient-based optimization algorithms:

$$\omega^{t+1} = \omega^t - \alpha \nabla \mathcal{L}(\omega^t) \tag{2.1}$$

where the weights ω are adjusted in the negative direction of the gradient of $\mathcal{L}$, with α representing the learning rate, which regulates the magnitude of each update. In most cases, weights are initialized randomly, and after each exposure to examples of data (a training step, represented by equation 2.1) they get closer

to convergence, i.e. the point at which the model parameters stop adjusting significantly. This iterative learning process is often referred to as *gradient descent*, and it is highly dependable on our ability to find good optimizers of highly non-convex loss functions. An important issue with gradient descent, in addition to speed, is the risk of incurring in local minima and saddle points of the gradient. An optimizer with good convergence should be able to avoid remaining stuck in local minima and eventually reach a global minima. Several gradient-descent optimization algorithms have been proposed, see Ruder (2017) for a complete overview.

2.2 Overfitting and the bias-variance tradeoff

When a model learns from a set of training data, the ultimate goal is that the algorithm will also perform well when exposed to new data that was not encountered during its learning phase. Overfitting occurs when the model parameters adjust too closely to the training data, learning examples "by hearth" instead of abstracting the relevant patterns. In other words, an overfit model unknowingly learned some of the random noise (unrepresentative variation) in the data as if it represented the underlying function. Consequently, it will have a weaker generalizing capability and perform poorly with unseen data. To evaluate whether a trained model overfits, one usually tests it on an independent set of data for which the labels are known. The opposite problem, *underfitting*, comes when a model's approximation of the function is too simplistic. Another central problem of supervised learning[1], intimately related to overfitting, is the bias-variance tradeoff. It can also be seen as a conceptual framework used to find the right balance between overfitting and underfitting. The bias error comes from erroneous assumptions in the learning algorithm. An example is presented in Fig. 2.2, where a linear model is used to approximate a set of points that evidently do not show a linear relationship. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting). The variance is an error from sensitivity to small fluctuations in the training set (e.g. figure 2.2). High variance can cause an algorithm to model the random noise in the training data, rather than the intended

[1] Supervised learning is the machine learning task of inferring a function from labeled training data.

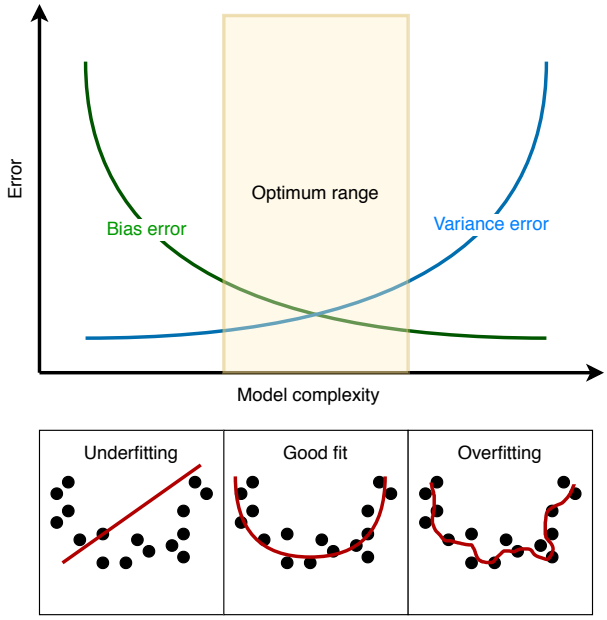

Fig. 2.2.: A simplified representation of the bias-variance tradeoff. Above, the change in a model's error based on its complexity; below, a visual representation of a model's fitting to a training dataset. Ideally, one wants a model that is complex enough to model the underlying function, but not too complex to learn the unrepresentative variation in the training data: this is the model that falls in the optimum range.

outputs (overfitting). The tradeoff is in that a model either captures the variability in its training data or it generalizes on unseen data, and it's impossible to do both simultaneously. The solution is to look for an optimum balance.

2.3 Artificial Neural Networks

2.3.1 Network architecture and hyperparameters

A neural network model is typically defined by its architecture and hyperparameters. The architecture essentially determines how the model neurons are connected with one another, and what happens during every transformation on each layer of the model. A well-known class of ANN is the fully-connected neural network, which consists of a series of layers that connect every neuron in one layer to every neuron

in the next layer. Since every layer only connects to the next without forming *cycles*[2], starting from the input layer and ending with the output layer, this kind of architecture is also characterised as *feed-forward*. The degree of complexity of such models is largely determined by the total number of parameters, which in turn depends on the number of hidden layers and the number of neurons in each layer. These and other elements that need to be set by the user (in contrast with the values of the model parameters or weights, which are often randomly initialized) are called *hyperparameters*. Other examples of hyperparameters include: the learning rate α, the activation functions, the number of epochs and the batch size. The learning rate, as already seen in section 2.1, determines the magnitude of corrections at each training step. A large α will result in a faster convergence, but with the risk of missing the right convergence pathways and incur in *exploding gradients*. On the other hand, a too small value can excessively slow down the training, with the risk of incurring in local minimas. Activation functions determine the output of each neuron for a given input or set of inputs, and are typically what allows ANN to model complex non-linear functions, since they provide a non-linear response for each neuron. A widely used activation function is the Rectified Linear Unit (ReLu), defined as $f(x) = max(0, x)$ where x is the input of a neuron. In other words, a neuron activates if the input is positive and deactivates if the input is negative. The number of epochs represents how many times the model sees the entire training dataset during the training phase. The batch size represents the number of examples that are used for each training step, i.e. the number of examples for which the gradient on the loss $\mathcal{L}$ is computer with respect to the model's weights.

2.3.2 Regularisation

Regularisation techniques are a variety of modifications applied to the learning algorithm used to prevent overfitting. Among the most popular is Dropout (Srivastava et al., 2014). The idea is to randomly deactivate some neurons during training, with a specific probability and repeatedly for each training iteration, such that the model optimizer won't update the weights associated with those neurons. Dropout

[2]When an ANN is designed such that layers form cycles it is called a Recurrent Neural Network, which is opposed to feed-forward architectures

can also be regarded as ensemble learning, as different explanatory pathways are combined to result in the final model. As it will be discussed in section 2.3.2, this aspect is of particular importance for probabilistic modelling. Another well-known regularisation technique is early-stopping (Prechelt, 1998). Fundamentally, one keeps track of the learning curve of the model during training, both for the training and validation dataset, and stops the training if the validation error stays the same or increases, while the training error continues to decrease.

3 Data and methods

3.1 Observational dataset

The observational data used in this study comes from multiple sources, as shown in Table A.1. The selected dataset consists of hourly mean observations of several meteorological parameters including wind speed, wind direction and sea-level pressure, and the hourly maximum for wind gust. The 739 measurement stations are distributed over the alpine area (see Fig. 3.1), covering several kinds of geomorphological settings, and for this study were considered observations over a four years period, ranging from April 2016 to April 2020. In total, excluding missing values, this adds up to roughly 20 millions wind observations. The measurement networks originally included a larger number of stations, but some of them were excluded after conducting a quality assessment that is described in Section 3.7. Additionally, we considered a set of automatic daily weather classification schemes introduced at MeteoSwiss by Weusthoff (2011), described in Table 3.1.

Tab. 3.1.: CAP9: a daily weather classification with 9 classes derived by a principal component analysis and subsequent clustering of ERA40 reanalysis, based on mean sea level pressure in the alpine region.

Wheather classification description	Code	Frequency
NorthEast, indifferent	0	0.23
West-SouthWest, cyclonic, flat pressure	1	0.16
Westerly flow over Northern Europe	2	0.13
East, indifferent	3	0.13
High Pressure over the Alps	4	0.11
North, cyclonic	5	0.09
West-SouthWest, cyclonic	6	0.07
High Pressure over Central Europe	7	0.05
Westerly flow over Southern Europe, cyclonic	8	0.03

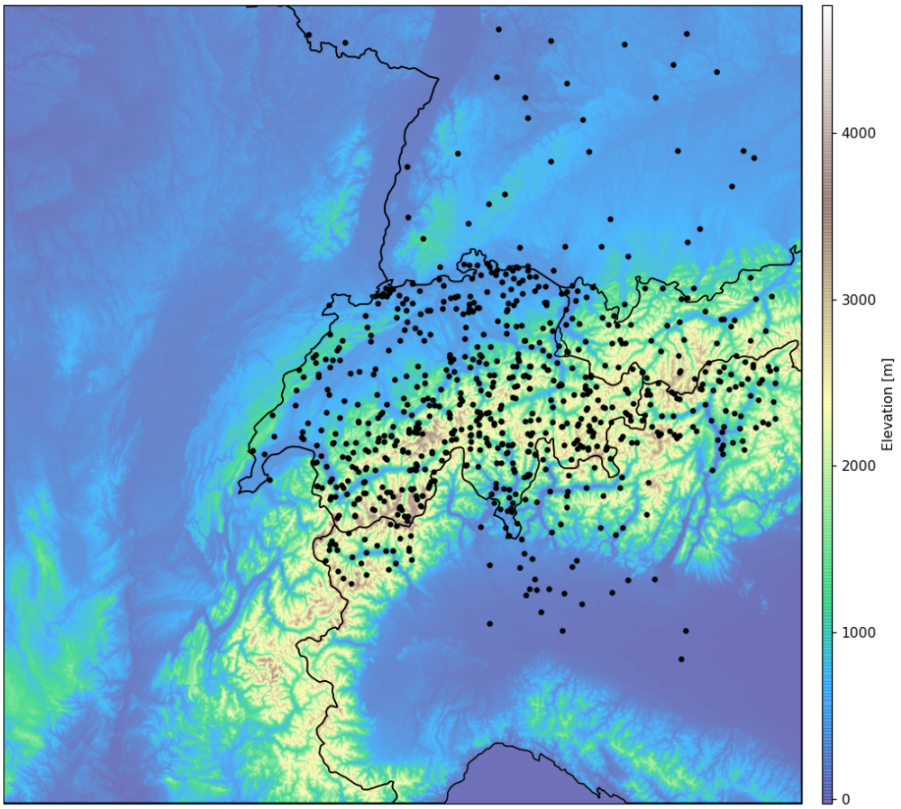

Fig. 3.1.: Location of all 739 selected stations over the Alpine area, represented with its topography.

3.2 NWP dataset

The NWP model used as input to train our model is COSMO-1, a state-of-the-art regional model operated by MeteoSwiss. The model runs with a deterministic and non-hydrostatic configuration, and has a high horizontal resolution of 1.1km. The archive of predictions covers about the same spatial and temporal domain of the observational dataset, and consists of hourly values of several meteorological parameters, including wind speed, wind gust and wind direction. In addition to wind related parameters, we considered the boundary layer height as a predictor for our model, as well as differences in pressure between specific locations (Lugano and Basel to account for North-South gradient, Geneva and Güttingen for West-East gradient).

As a baseline for the evaluation of our model we used COSMO-E, the 21 members ensemble configuration of COSMO model, that has an horizontal resolution of 2.1km. The choice of using COSMO-E instead of COSMO-1 was motivated by the need for consistency in the objective quantification of the model performance: deterministic and probabilistic forecasts are difficult to compare.

The best solution would have been to use COSMO-1E (the ensemble configuration of COSMO-1), but due to major adjustments on the model's physical parameterizations during the pre-operational phase, the homogeneity of the archived data, an important quality for a machine learning training dataset, had been compromised. The issue of frequent model changes, violating the assumption that the error characteristics remain constant over time, and the proposed solutions are discussed in Vannitsem, Bremnes, et al. (2020).

3.3 Topographical descriptors

The geomorphological setting of a location explains a lot of the sub-grid scale variability of surface wind speed. Therefore, we need a way to characterize the landscape configuration in a way that is meaningful with respect of the process of interest. Using a Digital Elevation Model (DEM) as a starting point, there are two approaches to derive meaningful representations of topography that are useful for the prediction of surface wind speed: to feed the raw DEM data to an ANN incorporating convolutional layers, which then automatically abstracts different levels of representation during the learning phase; to derive topographical descriptors manually, based on domain knowledge, performing what's known in machine learning as *feature engineering*. Both approaches were evaluated by Schär (2019), and the latter approach is shown to be preferable both in terms of forecast performance and computational efficiency. In addition, it facilitates the interpretation of the model using domain knowledge. A comprehensive set of topographical descriptors was considered in this study, namely: South-North derivative, East-West derivative, Slope, Aspect, Valley Norm and Direction, Ridge Norm and Direction, Topographic Position Index (TPI), Sx. As shown in Chapter 4, only a small subset of these topographical descriptors was included in our final model.

3.3.1 Directional Derivatives

The South-North and East-West derivatives are simply calculated using centered finite-difference formula as:

$$d_{NS}(x,y) \approx \frac{DEM(x,y+1) - DEM(x,y-1)}{2\Delta y}, \tag{3.1}$$

$$d_{EW}(x,y) \approx \frac{DEM(x+1,y) - DEM(x-2,y)}{2\Delta x}. \tag{3.2}$$

The different spatial scales where calculated by applying a Gaussian smoothing filter with the corresponding window size to the DEM before computations.

3.3.2 Slope

The Slope is defined as the magnitude of the DEM gradient at any given location and can be derived from the directional derivatives:

$$slope(x,y) = \sqrt{d_{EW}(x,y)^2 + d_{NS}(x,y)^2}. \tag{3.3}$$

3.3.3 Aspect

The Aspect is defined as the direction of the DEM gradient at any given location and can be derived from the directional derivatives:

$$aspect(x,y) = atan2\left(\frac{d_{EW}(x,y)}{d_{NS}(x,y)}\right). \tag{3.4}$$

3.3.4 Topographical Position Index (TPI)

The TPI is a simple metric used to describe landform types such as hilltops, exposed ridges, valley bottoms etc. (Weiss, 2001). It's defined as the altitudinal difference between a considered location and the mean elevation of its surroundings (which is defined here using the general expression of a convolution[1]):

$$TPI(x,y) = DEM(x,y) - \sum_i \sum_j M(i,j)DEM(x-i, y-j), \tag{3.5}$$

where M is a mean filter kernel of size $i \times j$. The extent of the convolution kernel around (x, y) determines the scale of the TPI.

3.3.5 Valley Index and Ridge Index

This Valley Index was proposed by Schär (2019) to describe valley shapes and their main orientation in an attempt to account for wind channeling effects. The derivation consists in convolving the DEM with valley-like kernels of varying size (to account for different valley widths) and shape (to account for different valley type, e.g. U-shaped or V-shapes). The kernels are applied to every pixel of the DEM using the Fast Fourier Transform, with varying orientations (0-360 degrees with 1 degree increment) and then combined to create the Valley Index.

$$ValleyIndex(x,y) = \sum_i \sum_j V(i,j)DEM(x-i, y-j), \tag{3.6}$$

where V is the valley-shaped kernel of size $i \times j$. Additionally, the magnitude of the Valley Index was multiplied by the sine and cosine of the valley orientation to result in two descriptors, one for each component of the valley orientation. The Ridge Index follows the same principle of the Valley Index, but the kernels are

[1] Convolution is the process of adding each element of a matrix to its local neighbors, weighted by a kernel. It is extensively used in image processing.

reversed in order to highlight ridges instead of valleys. The different spatial scales are determined by the size of the convolution kernel.

3.3.6 Sx

The Sx represents the maximum slope among all imaginary lines connecting a given pixel with all the ones lying in a specific direction and up to a maximum distance (Winstral et al., 2017). Sx is a proven wind-specific terrain parameterization capable of differentiating such slopes based on given wind directions, therefore adding flow-dependency to the model input space. The derivation of the Sx is formulated as:

$$Sx_{az,d_{max}}(x,y) = max\left(tan^{-1}\left(\frac{DEM(x_v,y_v) - DEM(x,y) + height}{[(x_v-x)^2 + (y_v-y)^2]^{0.5}}\right)\right), \quad (3.7)$$

where az is the azimuth of interest, d_{max} is the maximum distance (radius), (x, y) the considered pixel coordinates and (x_v, y_v) the set of all pixels coordinates lying in an area delimited by d_{max} and a cone centered around az. A fast Python routine was developed specifically for this task, which makes use of Bresenham's line algorithm to select (x_v, y_v). The height parameter was set to 10 meters, consistently with the standard height for wind measurements. The Sx was calculated every 5°, for a total of 72 azimuths ranging from 0° to 355°.

3.3.7 Model-DEM height difference

Although not purely based on high resolution DEM, an additional descriptor was considered that represents the difference in height of the topography used by the NWP model and the height of the high resolution DEM.

3.4 Temporal descriptors

The daily cycle is the main driver of thermal winds, and the annual cycle largely governs the frequency of occurrence of weather regimes that determine particular flow situation e.g. Bise, Foehn. To account for this, the models were provided with temporal descriptors. Since both the hour of the day and the day of the year are circular variables, they required an encoding using sine and cosine functions to express them in two components:

$$cos(h\frac{2\pi}{24}) \quad ; \quad sin(h\frac{2\pi}{24}), \tag{3.8}$$

$$cos(d\frac{2\pi}{365}) \quad ; \quad sin(d\frac{2\pi}{365}), \tag{3.9}$$

where h is the hour of the day and d is the day of the year.

3.5 Integration of real-time information

The key idea of nowcasting is to use all the latest available information, typically NWP forecasts and measurements (e.g. from station data or satellite imagery), to produce an accurate analysis of weather parameters and extrapolate a forecast up to a few hours ahead. The use of real-time measurements as predictors for our models raises two questions, considering that the model must be able to make prediction for ungauged locations[2]. First, which measurement is chosen as predictor at any given location and at a given time; second, how relevant that measurement is for the final prediction. These are common problems for spatial interpolation, where the goal is to use point sampled measurements to generate

[2]we designate as "ungauged locations" points in space where no measurements are available

spatially continuous data. To do this, nearly all methods share the same general estimation formula:

$$\hat{z}(x_0) = \sum_{i=1}^{n} \lambda_i z(x_i), \tag{3.10}$$

where $\hat{z}$ is the estimated value of the primary variable at the point of interest x_0, z is the observed value at the sampled point x_i, λ_i is the weight assigned to the sampled point, and n represents the number of sampled points used for the estimation. Methods differ in the way λ_i is computed. In the environmental sciences a large number of approaches have been proposed (Li and Heap, 2011), even in combination with machine learning algorithms, and these are shown to be often data-specific and variable-specific, i.e. there is not a single best solution but rather several methods tailored to different problems.

The interpolation of surface wind is a very difficult task, particularly in complex topography. While several examples exist (Li and Heap, 2011; Reinhardt and Samimi, 2018; Scheuerer and Möller, 2015), to the best of our knowledge this work is the first to consider the interpolation of measurements of surface wind with an hourly granularity and at very fine scales of down to 100m. Moreover, there is no established method developed in a flexible[3] and computationally efficient way that allows to interpolate real-time data in a statistically optimised approach. Nowcasting is also an important component of *seamless*[4] prediction systems, thus spatial and temporal consistency must be ensured during the transition from the analysis (when real-time information has a higher influence on the prediction) to longer lead-times. These premises lead us to think of interpolation not as an explicitely separate procedure, but rather an intrinsic component of the ANN post-processing model.

[3] A challenging aspect of nowcasting systems is that they must be able to deal with a varying availability of real-time data. Therefore, several statistical techniques that rely on complete time-series (e.g. Principal Component Analysis), despite being attractive for historical data interpolation, are not fit for operational use.

[4] The word "seamless" usually denotes the paradigm of unifying weather prediction systems and their components across all time scales.

3.5.1 Similarity Index

This study proposes a new technique that is inspired by a concept of geostatistical interpolation methods, the semi-variogram, to rapidly estimate a *climatological* λ between any pair of points in space. The method consists in calculating the correlation matrix of the target variable for several pairs of stations, and then train a model to predict the correlation of each pair based on the absolute difference of topographical descriptors and geographical coordinates for those locations. This is expressed as:

$$\lambda_{ij} = f(|X_i - X_j|), \tag{3.11}$$

where λ_{ij} is the Similarity Index between stations i and j, f is the deep learning model, X_i and X_j are the values of the topographical descriptors (and optionally geographical coordinates) at stations i and j respectively. In order to reduce the impact of spurious correlation, and as a way to isolate the correlation of local scale weather opposed to synoptic scale weather variability, only pairs of stations lying at a maximum distance of 30 km from each other were considered. After this selection, a final number of 10292 pairs was used to fit a model, where every station had on average 13 neighbours. In order to assess the benefit of using topographical descriptors, a *naïve* model was considered as a benchmark, which only considered geographical coordinates and height.

In practice, the Similarity Index is used as follows: for any prediction we calculate the Similarity Index between the target location and the 10 closest gauged locations, and then chose the gauged locations with the largest value. Next, the observed wind speed measurement from that station is included as predictor in the ANN model, along with the respective Similarity Index value and leadtime (i.e. the age of the measurements). Ideally, the ANN model is able to weigh the influence of the chosen wind speed measurement based on the Similarity Index and the leadtime: the larger value for the Similarity Index, the larger the influence. This method is similar to regression Kriging in the way it uses a regression model and additional covariates to generate a semi-variogram.

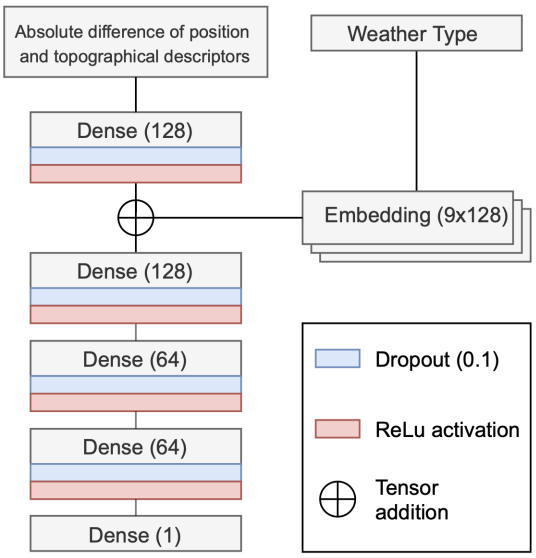

Fig. 3.2.: The Similarity Index model architecture.

An important aspect concerning the Similarity Index is that between gauged locations (where the value is 1) and the rest of the predicted values there is a "gap" in the distribution, which is also observable in Fig. 4.1. The cause is very simple: in the attempt to cover as much territory as possible, the measuring network was developed in a way that purposely avoids setting up weather stations that are very close to each other. This results in a small number of stations that show high correlations (close to 1) of wind speed, and consequently in a very unbalanced dataset where the occurrences of high values of Similarity Index are extremely rare. Our way to address this will be discussed in Section 3.8.2.

3.5.2 Stratification: a step towards flow-dependency

Ideally we want a model that is fully flow-dependent, i.e. that calculates the semi-variogram for every timestep, but that is also robust to noise. This is difficult to obtain. This problem is analogous to the bias-variance tradeoff, except in this case the model complexity can be translated to model *flow-dependency*. The analogy suggests that in this case too the best approach should be to find a good balance. Based on this rationale, we decided to stratify flow-dependency based on synoptic conditions, that is, we calculated a correlation matrix for different weather

types (the derivation of these classification is presented in Weusthoff (2011)), and included the weather classification index as a categorical non-ordinal predictor for the Similarity Index, using an embedding layer[5]. The ANN model architecture is shown in Fig. 3.2. The embedding layer encodes each of the 9 weather types into a tensor of 128 units, and depending on the input weather type it adds the corresponding tensor to the output of the first hidden layer of the model.

3.6 Probabilistic Models and Bayesian Neural Networks

This work was developed within an internal code base used at MeteoSwiss to facilitate research in post-processing using machine learning. The ANNs used in this project were developed using the python Deep Learning library Keras (Chollet, 2015), running on top of the machine learning platform Tensorflow[6]. Keras provides a simple application programming interface to build deep learning models, while Tensorflow facilitates the machine learning workflow with a comprehensive ecosystem of tools and resources. Additionally, Tensorflow Probability provides tools to build probabilistic models, e.g. allowing to have a Conditional Probability Distribution (CPD) as model output. Several PDFs are used to describe wind speed frequency distributions. Although none of them is able to generalize all wind regimes encountered in nature, some present clear advantages (Carta et al., 2009). In a case study conducted by Carta et al. (2009) well known distributions such as the Weibull or Gamma distribution explained >99% of variability for multiple stations. The Gamma distribution of the output y used in this study is defined as:

$$pdf(y;\alpha,\beta) = \frac{\beta^{\alpha} y^{\alpha-1} e^{-\beta y}}{\Gamma(\alpha)}, \tag{3.12}$$

[5]Embeddings are methods for learning vector representations of categorical data. They are most commonly used for working with textual data, because they capture some of the semantics of words.

[6]https://www.tensorflow.org/

where α and β represent the concentration and rate parameters of the distribution, respectively. In practice, these are the values of the nodes in the last hidden layer of the ANN, which are then fed to the output probabilistic layer as shown in Fig. 3.5.

3.6.1 Quantifying uncertainty

Being able to infer a CPD allows to estimate the *aleatoric uncertainty*, by adapting the shape of a distribution. This uncertainty is inherent to the process of interest and represents the variability that cannot be described by the input data of the model. In the case of wind speed, or for others meteorological parameters for that matter, is a consequence of the chaotic nature of the atmosphere and the measurement errors. In a general way, the predictive distribution for the output y in such conditions would be:

$$p(y|x, w), \tag{3.13}$$

where x represents the input data and w the model parameters that eventually define α and β in our case. Unfortunately, ANN models that use this approach tend to be miscalibrated. The predictive distributions are overconfident (i.e. under-dispersive), therefore worsening the reliability of a forecast. The reason for this deficiency is that conventional methods for probabilistic modelling ignore another kind of uncertainty: the *epistemic uncertainty*, also referred to as *model uncertainty*. This is related to the erroneous assumption that a model (i.e. its parameters w) is completely determined by a finite dataset. Instead, one must recognize the uncertainty of the model itself, represented by its posterior probability $p(w|D)$ where D is the data used for training. That's where Bayesian Neural Networks are introduced, with a new way to formulate the predictive distribution:

$$p(y|x, D) = \int p(y|x, w)p(w, D)dw. \tag{3.14}$$

Equation 3.14 represents the Bayesian Model Average (BMA). Rather than bet everything on a single hypothesis (a single model with fixed w parameters), we use every possible setting of parameters w, weighted by their posterior probabilities. Note that in this formulation the probability is not conditioned on w but on D, thus the idea that the model parameters have been *marginalized*. Finding an accurate and fast way to approximate the BMA integral has become an important subject of research in Deep Learning, and several approaches have been proposed. For more theoretical background the reader may refer to e.g. Wilson (2020), Wang and Yeung (2020), Dürr et al. (2020), and Jospin et al. (2020).

3.6.2 Monte Carlo Dropout

A widely used technique for a Bayesian approximation is Monte Carlo (MC) Dropout (Gal and Ghahramani, 2016). As mentioned in Chapter 2, Dropout is a regularisation technique used to prevent overfitting during training, whereby weights are randomly deactivated. The idea behind MC Dropout is to also activate dropout during inference. Then, for a given input x, one makes several predictions where each prediction results from a slightly different version of the model. Specifically, one predicts for the same input x T-times a CPD corresponding to a combination of weights w_i, or in other words one takes samples $y \sim p(y|x, D)$ of T different configurations of w (whereas in a non-Bayesian approach we would simply create an ensemble by sampling from a model with fixed w). Then, the dropout predictions are combined to a Bayesian predictive distribution:

$$p(y|x, D) = \frac{1}{T} \sum_{t=1}^{T} p(y|x, w_t) \tag{3.15}$$

that is shown to be an empirical approximation of Equation 3.14.

3.6.3 Assessment of the predictive performance

To assess the quality of a probabilistic forecast, one must assign a numerical score based on the predictive distribution and the value that materializes. Gneiting, Balabdaoui, et al. (2007) contended that the goal of probabilistic forecasting is to "maximize the sharpness of the predictive distributions subject to calibration". Sharpness refers to the concentration of the predictive distributions and is a property of the forecasts only. Calibration refers to the statistical consistency between the forecast distributions and the observations and is a joint property of the predictions and the events that materialize. Further studies have formalized this framework for forecast verification by linking it to decision theory, specifically *proper scoring* rules (Gneiting and Raftery, 2007; Gneiting and Katzfuss, 2014) and identified metrics with the most desirable properties. Among these an especially attractive metric is the Continuous Ranked Probability Score (CRPS) (Matheson and Winkler, 1976), chosen for this study. The CRPS addresses both sharpness and calibration, is negatively-oriented and can be interpreted as a generalised version of the Mean Absolute Error for the case of probabilistic forecasts (Gneiting and Raftery, 2007), and is therefore expressed in the same units as the target value. The CRPS is defined as:

$$CRPS(F, y) = -\int_{-\infty}^{+\infty} (F(y) - \mathbb{1}(x \geq y))^2 dx \tag{3.16}$$

where F is the cumulative distribution of the predicted distribution and y is the materialized value. The equation therefore corresponds to the integral of the Brier score along all real-valued thresholds x. Gneiting and Raftery (2007) gave an alternative formulation more suited for computation in the case of ensemble predictions and showed that:

$$CRPS(F, y) = E_F|\hat{Y} - y| - \frac{1}{2}E_F|\hat{Y} - \hat{Y}'| \tag{3.17}$$

where $\hat{Y}$ and $\hat{Y}'$ denote independent random variables drawn from the forecast distribution associated with F, and E_F denotes the expectation value under F, that

in our case was evaluated from 100 samples. The CRPS formulated as in 3.17 is used in this study as loss function $\mathcal{L}$ during training.

To qualitatively assess the calibration of a probabilistic forecast, Probability Integral Transform (PIT) histograms are an appropriate tool that complement the CRPS. The use of the PIT is discussed in detail in Gneiting, Balabdaoui, et al. (2007). Considering a probabilistic forecast and materialized observation pair (F, y), the PIT is defined as:

$$PIT = F(y). \tag{3.18}$$

If the forecast is perfectly calibrated, then the PIT values follow a standard uniform distribution. This is equivalent as saying that, considering the random variable y, y is indeed drawn from the predicted distribution F. One should note however, that the uniformity of the distribution is a necessary but not sufficient condition for the forecast to be perfect, as shown by Hamill (01 Mar. 2001). In practice, when dealing with ensemble forecasts, Eq. 3.18 can also be expressed as:

$$PIT = \frac{1}{M} \sum_{m=1}^{M} \mathbb{1}(\hat{Y} \leq y). \tag{3.19}$$

where M is the number of members of the ensemble, $\hat{Y}$ is a sample drawn from the predicted CDF and y is the observed value.

3.7 Preprocessing

In machine learning, preprocessing is an important step where the raw data is prepared to be used by the model. This typically involves a quality control (e.g. outlier detection, checking homogeneity of a time series), transformations applied to the data and the creation of independent subsets of data.

3.7.1 Dataset preparation and transformation

The observational dataset used in this study was filtered with the following criteria: (a) exclusion of stations with known bad quality measurements, (b) exclusion of WSL stations located in forests, (c) exclusion of IMIS snow stations, (d) exclusion of individual suspicious measurements. Steps (a) and (c) were applied under advise from domain experts. Step (b) was applied considering land-cover information is not used as predictor in the post-processing model. Step (d) was applied by excluding sequences of fixed value measurements, which may be artifacts resulting from software or hardware errors (e.g. frozen measurement device). Missing observations for gaps of up to three hours were filled by linearly interpolating in time, and finally an additional cleansing was applied by excluding all time references with missing values either from the NWP or the observational dataset. In addition to the collection of wind speed data itself, part of the dataset preparation is also to collect meta-data about each station. In this sense, a particularly important information is the height above the ground of an anemometer, which varied greatly in our dataset (from a minimum of 2 meters to a maximum of 62 meters) and influences the observed wind speed. Missing values were set to 10 meters, as this is the default for most weather stations.

Data scaling is a recommended pre-processing step when working with artificial neural networks. One type of scaling is standardization, which involves transforming the distribution of values of a dataset so that the mean of observed values is 0 and the standard deviation is 1.

$$x_{standardized} = \frac{x - x_{mean}}{x_{std}} \tag{3.20}$$

where x is the raw value, x_{mean} and x_{std} are the mean and standard deviations of the training dataset respectively. It is important to make sure that the new input features from independent datasets are always scaled with respect to the training dataset.

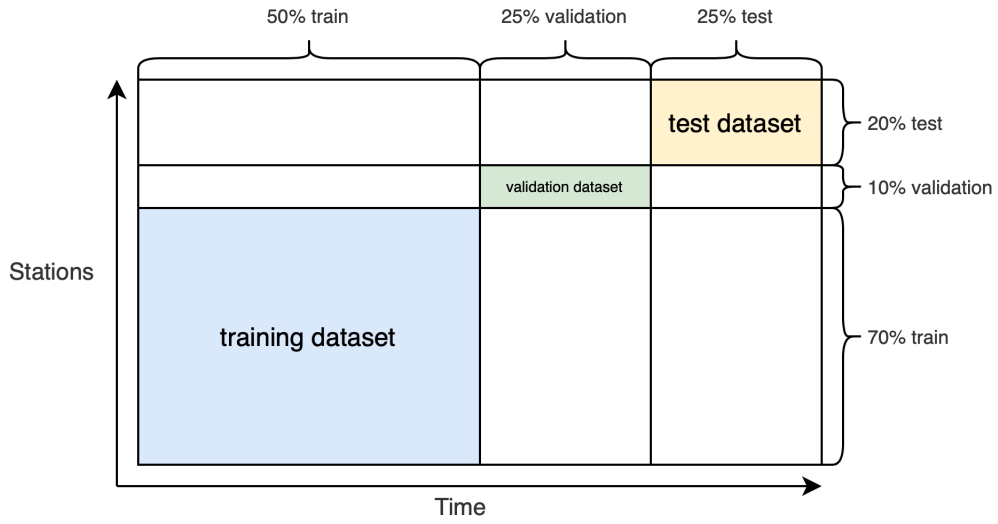

Fig. 3.3.: Dataset split along spatial and temporal dimension. Split for stations is random, split for time is sequential.

3.7.2 Data-split

In order to assess the generalisation capability of a model, a three-way split was applied to the dataset, resulting in training, validation and test independent datasets. The split was applied along both spatial (the stations) and temporal dimensions, as shown in figure 3.3. This allows to evaluate both the spatial and temporal generalisation. The split was randomised in the case of stations, and sequential for time references. The sequential split in time is necessary because we must ensure that no test sample is too close to a training sample, as this would violate the independence of the datasets.

3.8 Models

3.8.1 Baseline

The baseline model used for our evaluation is simply the COSMO-E nearest neighbor grid point around each measurement station.

3.8.2 Post-processing model (PP) and post-processing-nowcasting model (PP-NC)

A fully connected sequential model, known as the Multi Layer Perceptron (MLP) was considered in this study. This architecture is fairly simple to implement and is a proven solution for many regression and classification tasks in deep learning. A Rectified Linear Unit activation function was used and dropout layers were applied after every hidden layer, using a 50% rate. The architecture is displayed in figure 3.4. We used the same architecture to train two different models: a post-processing model, and a post-processing-nowcasting model that also includes real-time observations as predictors. For a matter of consistency, the two models where trained under the exact same conditions in terms of hyperparameters and dataset split. Both models were trained using a custom batch generator that allowed us to specify the number of different stations, reference times (i.e. the time that defines the start of a model run) and leadtimes in each batch. After some trials, we opted for batches composed of 100 stations, 100 reference times and 50 leadtimes. Although a more systematic way to tune this hyperparameters would have been preferable, it was out of the scope of this work. We used the Adam optimization algorithm and specified a learning rate of 0.001. In addition to dropout as a part of the models architecture, we implemented early stopping as a regularisation technique.

For the PP-NC model, an important aspect was to learn how to treat high values of Similarity Index. For the reasons explained in 3.5.1, namely the under-representation of high values, this was a challenge. Although in our dataset very few samples of high Similarity Index exist, in the real world, when we consider all locations in space and not just those where stations are located, these are actually much more common (see e.g. Fig. 4.2). The question is: how do we counteract this unbalance? We have found that a relatively simple solution was to over-sample data points with Similarity Index equal to 1., in other words examples where the observation of the station itself is used (we refer to them as "gauged" points or stations). We have tried several options and finally used a proportion of 30% of gauged stations in our training dataset, which resulted in a good treatment of high values of Similarity Index without detrimental effects on low values.

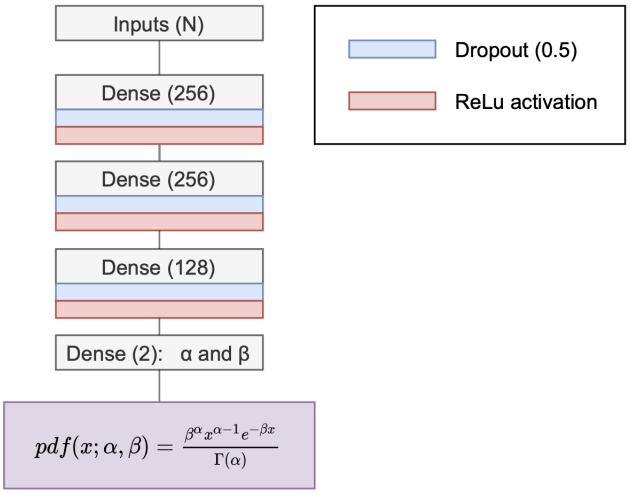

Fig. 3.4.: MLP model architecture used for the PP and PP-NC models.

3.9 Interpretability

3.9.1 SHapley Additive exPlanations

For the interpretation of the model, the SHAP (SHapley Additive exPlanations) framework (Lundberg and Lee, 2017) was considered for this study. SHAP uses a coalitional game-theoretic approach to estimate the contribution of each feature to the model output, i.e. it provides a fast way to estimate Shapley values (Lundberg and Lee, 2017). In this framework, the different predictors can be regarded as players in a coalition, and Shapley value determine how to fairly distribute the "payout", or prediction, among the players. SHAP has unified other methods for model explanations, such as LIME (Ribeiro et al., 2016) or DeepLIFT (Shrikumar et al., 2019), under a newly defined class of methods called *additive feature attribution method*. It is important to note that this kind of model explanations alone does not necessarily represent the process of interest, but rather serve as a tool for knowledge-based interpretation. Just like correlation does not imply causation, the impact of a feature on a prediction does not imply a direct physical cause, because explanations that we compute with SHAP are *true to the model*, and the model is unaware of physical relationships (unless we enforce them somehow). In other words, we are not able to strictly determine the true impact of a predictor

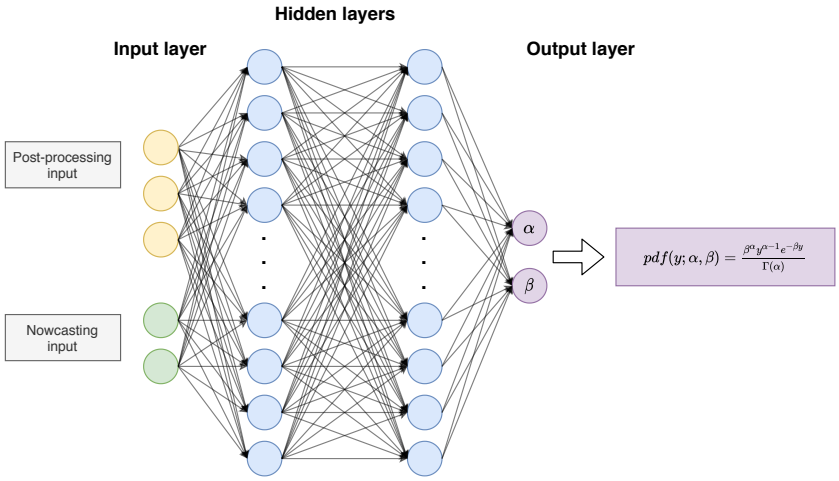

Fig. 3.5.: A simplified representation of the ANN used to make probabilistic predictions of wind speed. The two nodes of the last layer are the parameters α and β of the predicted distribution of y.

for the occurrence of a given target value: we are only able to explain how the model reaches its conclusion. This implies that counter-intuitive explanations can be observed, particularly in presence of multicollinearity. This question is discussed in more detail by Chen et al. (2020).

SHAP comes with a few interesting features: we get contrastive explanations, i.e. every individual prediction is compared with the average prediction. In other words, the SHAP values tell us how predictors are "pushing" a single prediction in one direction or the other with respect to an average of many predictions. While several other methods are generally limited to providing information about feature importance, with SHAP we are able to explain individual predictions. The framework also provides a useful set of visualisations to get insights about the role of each predictor[7]. For a matter of simplicity, the SHAP values were computed in this study only for the mean of the output CPD, therefore ignoring the uncertainty of the prediction.

[7] see https://github.com/slundberg/shap for some examples

4 Results and discussion

4.1 Similarity Index

4.1.1 Model performance

In this section we analyse the results obtained with the Similarity Index model, and compare a naïve solution with the final model. Figure 4.1 shows the predictions against the true correlations for an independent test dataset, for both the naïve model and the final model. Our best model could make predictions with a mean absolute error of 0.092, and the goodness of fit is also validated by an R-squared value of 0.823. Overall, the final model is capable of approximating the correlation of wind speed measurements between two locations based on their relative geographical and geomorphological setting. The use of the relative geomorphological setting in addition to the distance in terms of geographical coordinates (x, y, z) brings a significant improvement. This is consistent with the assumption that wind variability in complex topography is highly related to the surrounding topography of a location. Despite the good results for most samples, there are still substantial errors in few cases. However, a visual inspection could determine that the error distribution is close to Gaussian. This indicates that there is no systematic source of error and that it is random, likely due to unknown factors influencing the true correlation and/or to errors in its estimation.

A possible evidence for the latter is the fact that the MAE (shown in Table 4.1) is higher for less frequent weather classifications, for which correlations are less robust. To further evaluate the fitness of the model, we analysed the model SHAP explanation and inspected some examples of the Similarity Index computed over a region in the northern Alps.

Tab. 4.1.: MAE for each CAP9 weather classification.

CAP9 code	0	1	2	3	4	5	6	7	8
MAE	0.07	0.09	0.09	0.09	0.09	0.08	0.10	0.13	0.10

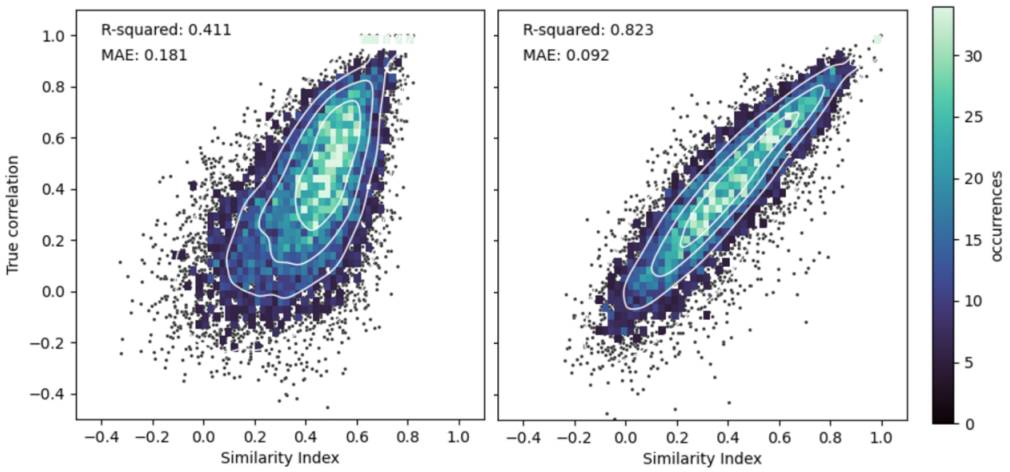

Fig. 4.1.: Predicted Similarity Index against the true correlation for station pairs of an independent test dataset, for a naïve model (left) and the final model (right).

4.1.2 Interpretation of the results

The analysis of SHAP values highlights the importance of the difference in elevation (DEM), as well as the TPI at a 500 meters scale. Our interpretation is that elevation is a main factor to determine the local weather regimes to which a location is subject, while the TPI at this particular scale is the most useful feature to distinguish between sheltered or exposed locations. The weather type also shows a significant effect on the overall prediction of the Similarity Index. While a characterisation of the contribution of each specific weather type is difficult with the available data, we believe a high degree distinction can be made between situations with strong and weak synoptic forcing. As shown in Figure 4.3, the former and the latter have respectively a positive and negative average impact to the final model output. This difference is likely due to the scale at which atmospheric dynamics take place, as well as their magnitude. For instance, in a situation with strong large scale advection, or during an event such as the passage of a cold front, wind speed varies similarly across large regions and different geomorphological settings. On the contrary, in the absence of large-scale or meso-scale dynamic systems we see a

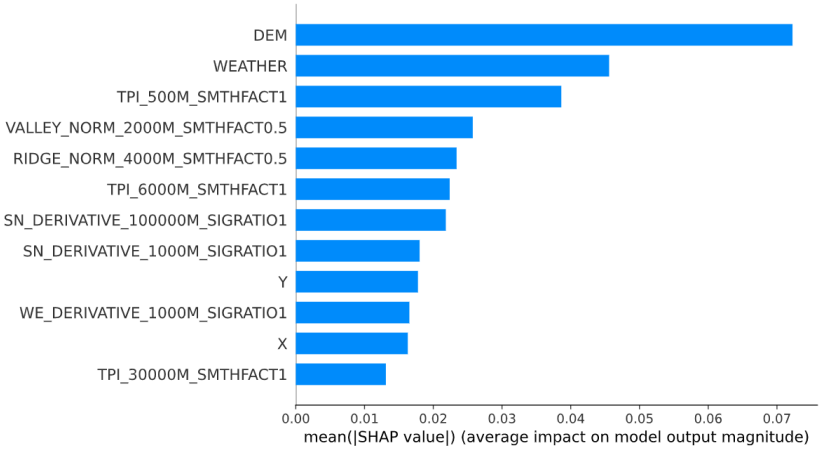

Fig. 4.2.: Mean absolute impact for each predictor of the Similarity Index model.

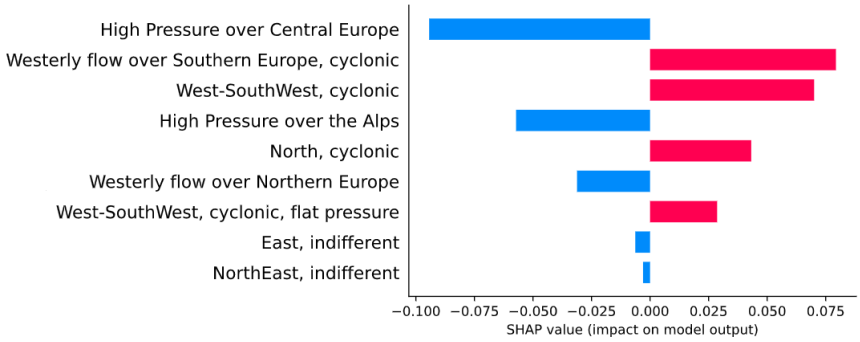

Fig. 4.3.: Average impact of each weather type classification (CAP9) on the predicted Similarity Index. Weather situations with strong synoptic forcing (such as low pressure systems or strong large scale advection) have a positive impact on the final output, while situations with weak synoptic forcing (such as high pressure systems) have a negative impact.

prevalence of small-scale weather regimes (e.g. the diurnal cycle) and localized effects of topography (e.g. crest speedups) in determining wind speed variability.

Figure 4.4 shows an example of the Similarity Index for a region of central Switzerland (see A.2), calculated with respect to the weather station (labeled "PIL") located on the Pilatus mountain massif at 2105 meters above sea level, for all weather types. The spatial distribution reflects the importance of predictors: the difference in elevation determines most of the variability, and topographical descriptors highlight specific features such as ridges, crests and narrow valleys. Moreover, it is evident how the model responds differently to weak and strong synoptic situations,

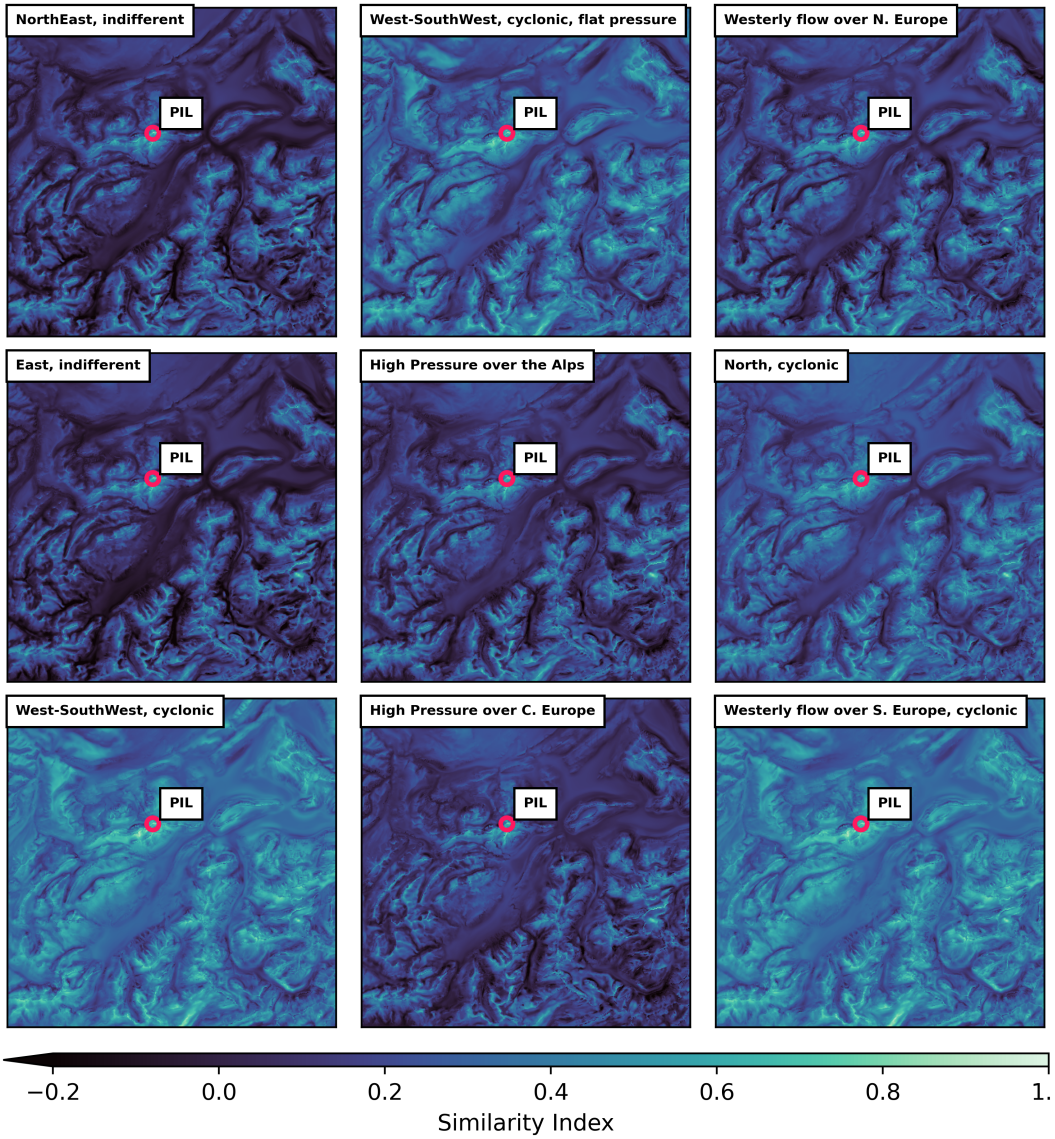

Fig. 4.4.: Similarity Index with respect to the weather station located on the Pilatus mountain massif at 2105 meters above sea level, for all CAP9 weather classification codes.

for example in CAP9 classification codes 7 (High pressure over Central Europe) and 8 (Westerly flow over Southern Europe). Interestingly, the difference between the two fields is stronger at low altitudes, but values are comparable for elevated locations and specifically mountain crests. Another example is shown in B.1 for a station located in the valley.

4.1.3 Representativeness of the measurement network

An interesting aspect of the Similarity Index is the ability to quantitatively assess how well wind speed at a given location is explained by the measuring network. In our application, for each target location we use the real-time wind speed measured from the station with the largest Similarity Index. It is therefore interesting to look at the maximum value of Similarity Index for each grid point in the area of study, to get an idea of which areas are going to benefit more from the proposed technique. This is shown in Fig. 4.5, where the maximum values have been averaged over all weather types. It is clear that regions of flat topography and valley bottoms are the most well represented, but we also observe large values on crests at high altitudes. In contrast, small values are generally observed on slopes. This result corresponds to our expectations because it reflects the geomorphological setting of stations in the study regions, that are either located on crests (e.g. SLF stations) or flat topography but not on slopes, which are therefore not well represented. In general, this result is expected for the whole network used in this study, because very few stations are located on slopes. For this very reason we also expect that the predicted values of Similarity Index at these locations are the most uncertain, because it is where the model relies the most on extrapolation rather than known examples. This aspect ties in on a broader discussion about the paradigms of weather monitoring. Traditionally, standardized and homogeneous conditions have been preferred in the design of monitoring systems. While this certainly makes it easier to analyse and compare historical time series, it has a detrimental effect on use cases that value a well-balanced diversity of conditions, such as training a deep learning model with generalizing capabilities. Ideally, all kinds of geomorphological settings should be represented, with varying conditions in terms

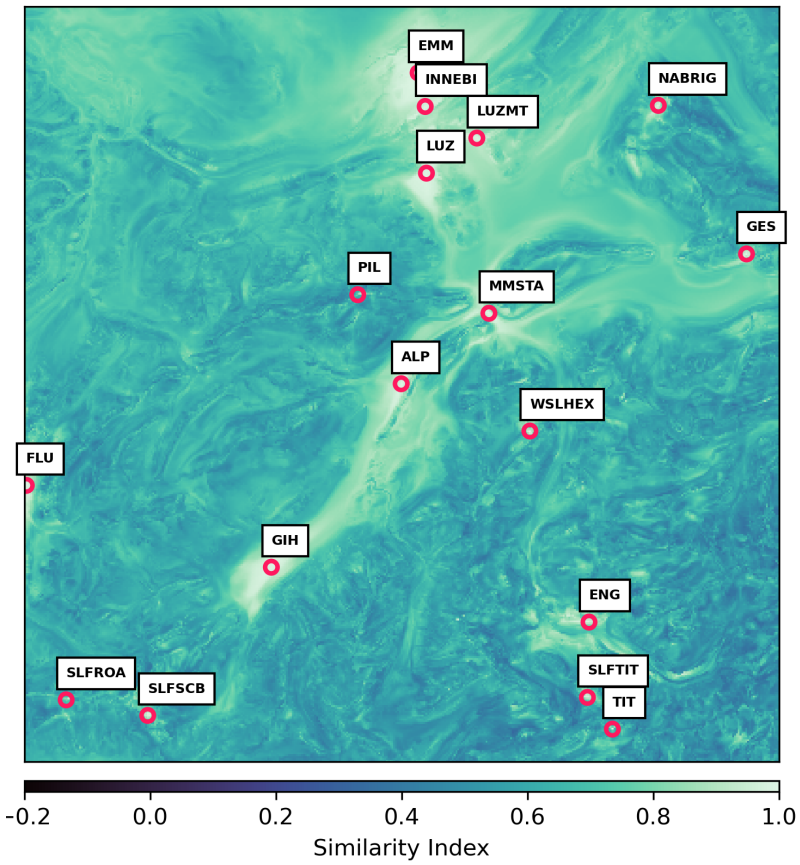

Fig. 4.5.: Maximum Similarity Index chosen among all reference stations, averaged over all CAP9 weather types.

of land cover as well, provided that meta-information is available. For instance, it would be interesting to account for the effects of terrain roughness.

Overall, we believe the Similarity Index has the potential to help in the development of a monitoring network, as it provides all the information needed to identify optimal locations for the installation of new stations. Nevertheless, this remains a non-trivial task that requires to evaluate all contributions of existing and new potential weather stations simultaneously. Moreover, one might chose to combine the results with additional information (e.g. density of population) based on the scope of the analysis.

4.2 Post-processing and nowcasting

This section presents the results for the wind speed post-processing and nowcasting model. First, we will compare the performance of different models. We will consider COSMO-E as a baseline, a normal post-processing model (PP) and a post-processing-nowcasting model (PP-NC). For our assessment we will use the CRPS as our objective function, and PIT histograms to qualitatively discuss the reliability (calibration) of our models. Additionally, we will discuss a few examples of predictions. We will then focus on aspects specifically associated with the use of real-time measurements and the role of the Similarity Index. A section will be dedicated to model interpretation, where we will analyse SHAP values to get insights into our predictors while also discussing whether or not the model behaves consistently with our expectations based on prior knowledge.

4.2.1 Models comparison

For this analysis, all metrics were evaluated on a dataset of unseen timesteps and unseen stations (see Section 3.7). Additionally, for the PP-NC model, we considered two cases: one for which the real-time measurement of the target station itself is used (from now on referred to as "PP-NC-gauged") and another that only uses real-time measurements from other stations. The latter is analogous to performing cross-validation for interpolation procedures, as it gives an indication of how the model performs in locations that are further away from weather stations. In order to avoid any confusion we want to stress that PP-NC and PP-NC-gauged are the same model, only evaluated in different conditions.

Figure 4.6 shows the wind speed CRPS of unseen stations calculated for leadtimes up to 10 hours. Compared to our baseline, all models significantly improve wind speed forecasts. As expected, PP-NC-gauged shows a sharp decrease in forecast errors for shorter leadtimes. This suggests that the model is able to incorporate real-time information and progressively rely on post-processing for longer leadtimes in a seamless way. In other words, the procedure commonly used in nowcasting systems known as *blending* is an intrinsic part of the model. A similar decrease, although much smaller in magnitude, is observed for PP-NC predictions

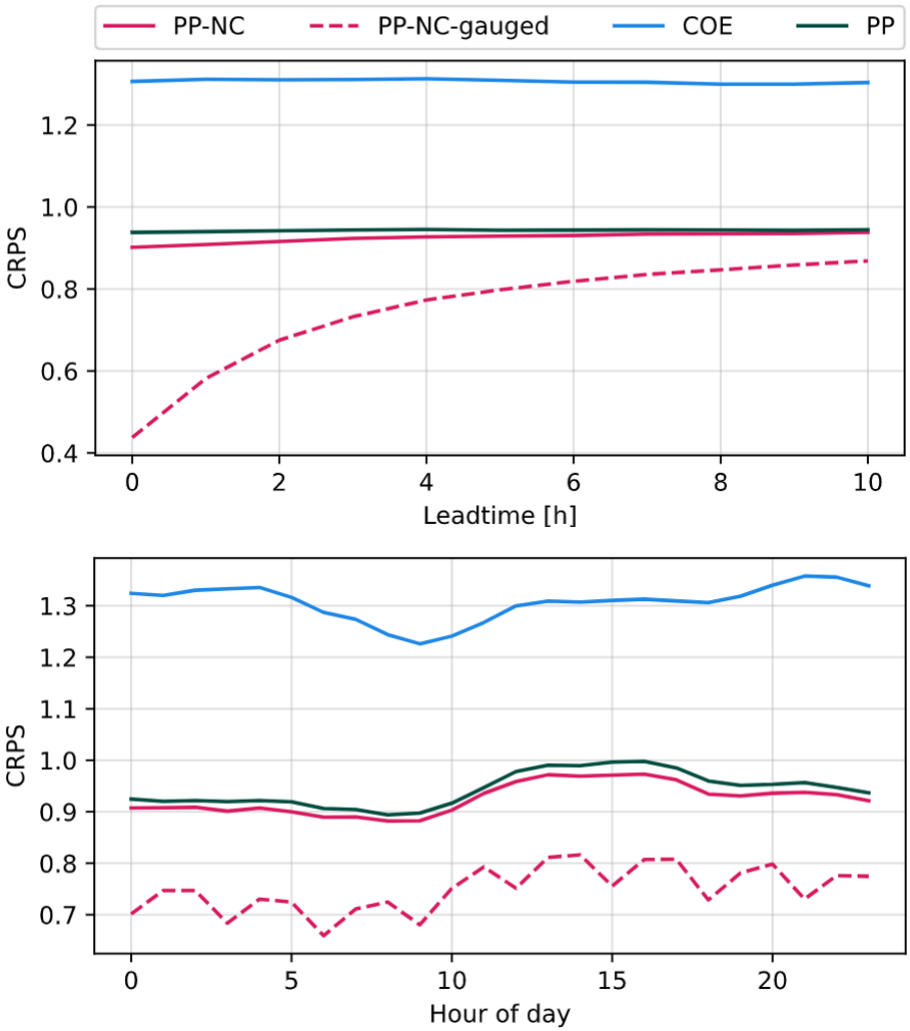

Fig. 4.6.: Top: mean CRPS from analysis time (leadtime = 0 h) up to 10h leadtime. Bottom: mean CRPS for each hour of the day.

at ungauged locations. This indicates that the use of real-time information, as determined by the Similarity Index, is effective even at distant locations. The bottom plot also presents the mean CRPS for each hour of the day. Note that for PP-NC-gauged the values oscillate because we evaluated model runs every 3 h. We observe a lower improvement of post-processed forecasts during the afternoon. This deficit in performance may be due to the

PIT histograms for the considered models are presented in Fig. 4.7. The dashed black line represents a perfectly calibrated forecast, for which PIT values would be uniformly distributed (since we consider 10 bins for our histograms, that

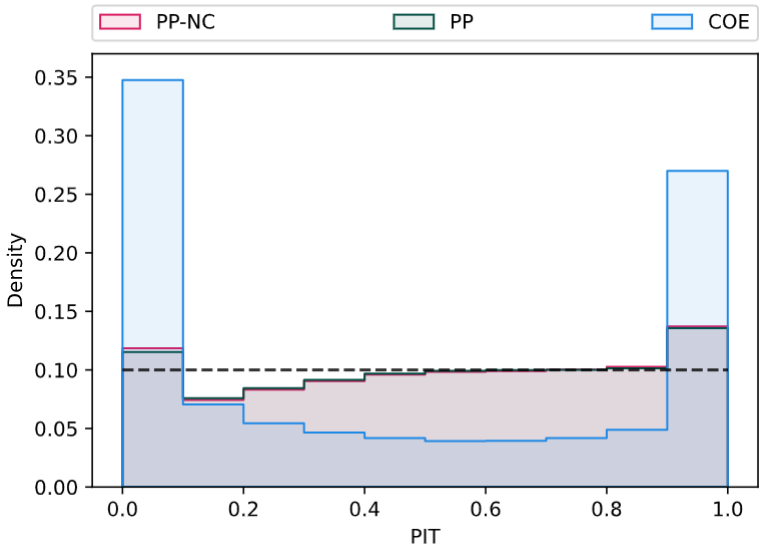

Fig. 4.7.: PIT histograms. The dashed black line represents a uniform distribution, that would be resulting from a perfectly calibrated forecast.

is equivalent to a uniform probability density of 0.1). The shape of the PIT histograms help us understand the nature of the model errors. The baseline model, represented in blue, is highly overconfident (underdispersive), missing occurrences of both extremely low and high wind speed. The slight asymmetry of the distribution towards low PIT values also implies a climatological overestimation of wind speed. Both PP and PP-NC models improve the reliability of wind speed forecasts considerably, and the difference between the two is barely noticeable. The rightmost bin indicates that some extreme wind speed values are missed by the model forecasts. The distribution of wind speed for completely missed high wind speed events (i.e. when all members of the ensemble are lower than the realized wind speed and the PIT value is equal to 1) is shown in B.4. We believe that these missed occurrences are in part due to a well known problem of regression tasks, where minimizing metrics such as the MAE (or the CRPS in our case) often leads models to predict values that are close to the mean. This unwanted effect can also be attributed to the training dataset being unbalanced (wind extremes, which are arguably the most interesting aspect of wind forecasting, are very rare). Additionally, another part of this lack of calibration could be attributed to highly unpredictable events such as thunderstorms resulting from localized convection. The left half of the histograms present an unusual shape. On one hand, the asymmetry indicates a climatological bias towards underestimation of wind speed,

but at the same time the leftmost bin is resulting from a large number of missed low values of wind speed. This is likely the result of a combination of factors that are difficult to identify. It is helpful to understand what contributes to the higher density of extremely low PIT values. Let us take PIT values equal to zero for instance, which occur when the realized wind speed is lower than all predicted ensemble members sampled from the CPD. Of the 50'951 occurrences, 73% resulted from missed forecasts of wind speed equal or below 0.2 m/s (this threshold is commonly used to define "calm" wind situations) and 90% for wind speed equal or below 1 m/s. The full distribution of wind speed is shown in B.5. It is also worth noting that wind speed values of this magnitude are often susceptible of measuring errors due to deterioration of cup anemometers, specifically they may underestimate weak winds (Pindado et al., 2014). This would introduce some unpredictable noise in our dataset, which in turn could partly explain the poor performance of the model.

In order to compare the baseline and the PP-NC model for actual predictions, while also looking at observed values, we used meteograms. We present probabilistic forecasts with median values of the ensemble represented by solid or dashed lines and inter-quantile ranges (middle 50% and 90%) shown as shaded areas.

4.2.2 Focus on nowcasting

In this section we discuss the use of real-time information by the PP-NC model. This includes a comparison of the model performance for ungauged and gauged targets, as well as a focus on the analysis (i.e. predictions for 0h lead time) of wind speed. Envisaging future developments, the goal here is also to provide an objective framework to evaluate this kind of deep learning models for nowcasting.

Sources of error

As already presented in Fig. 4.6, the CRPS decreases sharply for target locations where measurements are available. We deduce that, when the input real time observation is accompanied by a high Similarity Index, the model gives more importance to the observation. The CRPS reaches a minimum of 0.42 at 0h lead

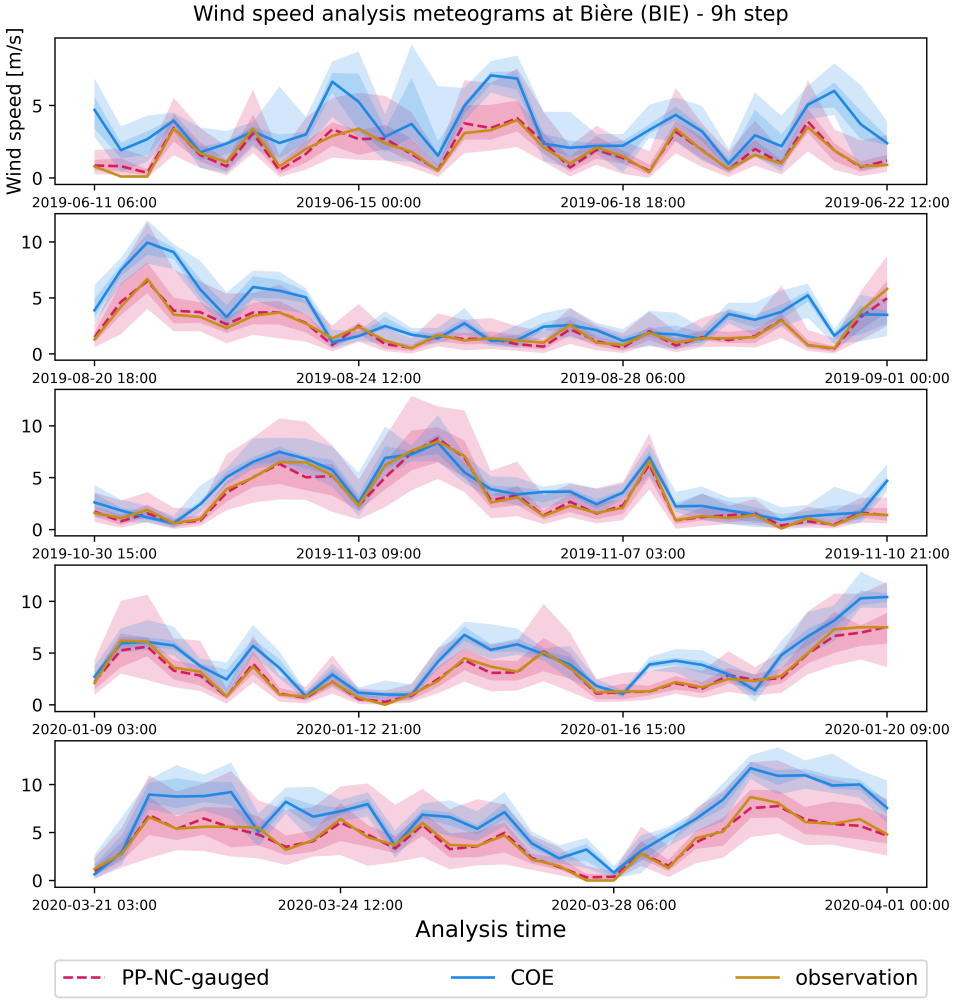

Fig. 4.8.: Examples of analysis meteograms for wind speed at Bière (BIE), using the measurements at the station itself.

time. Inevitably, this raises the question: if we are using the target wind speed measurement itself as predictor, why is the error not *zero* at 0h leadtime? Before we discuss the possible answers, it is useful to better understand the nature of this remaining error. To do that, PIT histograms come again to rescue. Figure 4.9 shows the PIT histogram for the PP-NC model at gauged locations, for predictions at 0h leadtime. PIT values were calculated on the independent dataset with unseen stations. Since the higher density occurs for central bins, it is evident that the forecast is overdispersive. That is, the average spread of the predicted ensemble is too large. Therefore, we can affirm that a significant part of the CRPS is due to an over-quantification of the uncertainty, rather than just systematic and conditional

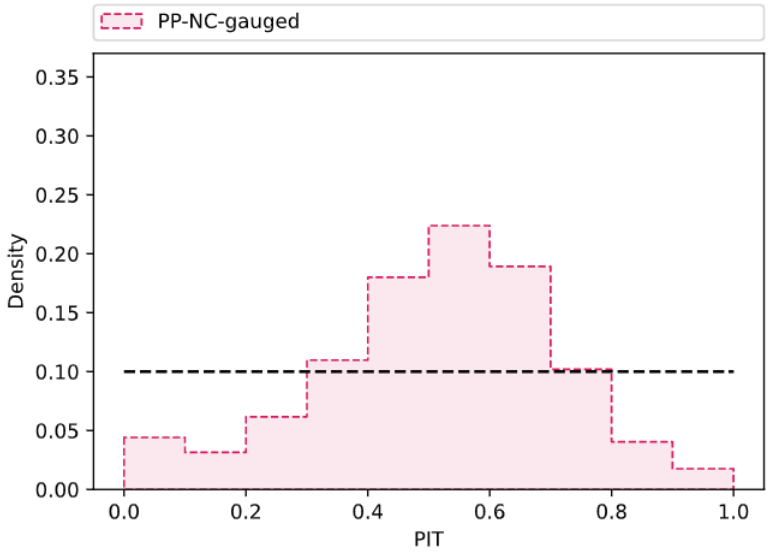

Fig. 4.9.: PIT histogram for the PP-NC model at gauged target locations, for predictions at 0h leadtime (analysis). The ∩-shape of the histogram indicates that the forecast is overdispersive.

biases. This can be further verified by looking at meteograms of the wind speed for the analysis. An example is presented in Fig. 4.8. The median of the PP-NC ensemble prediction, shown as a dashed line, is consistently very close to the observed values. Despite that, the spread of the ensemble remains relatively large, particularly for stronger winds. In B.6 we present the meteograms for the same predictions without using the observation of the station itself.

Ideally, at gauged locations (for which values of Similarity Index are equal to 1) and for 0h leadtimes, the model should only rely on the observations to predict a virtually infinitely sharp ensemble (i.e. a deterministic forecast) that coincides with the observation itself. This is not the case. We deduce from these arguments that the model has problems in using the real time observations correctly according to the leadtime, and this can be shown explicitely by aid of model explanations using SHAP values. Before we continue, let us remember an aspect that will help make sense of SHAP values. As already mentioned in Section 3.9, with SHAP we get contrastive explanations. This means that all model predictions are compared to an *average prediction* (also called baseline), that is calculated from a background sample (in our case it consists of 1000 examples), and SHAP values represent the contribution of each feature to get from the baseline to the actual model output.

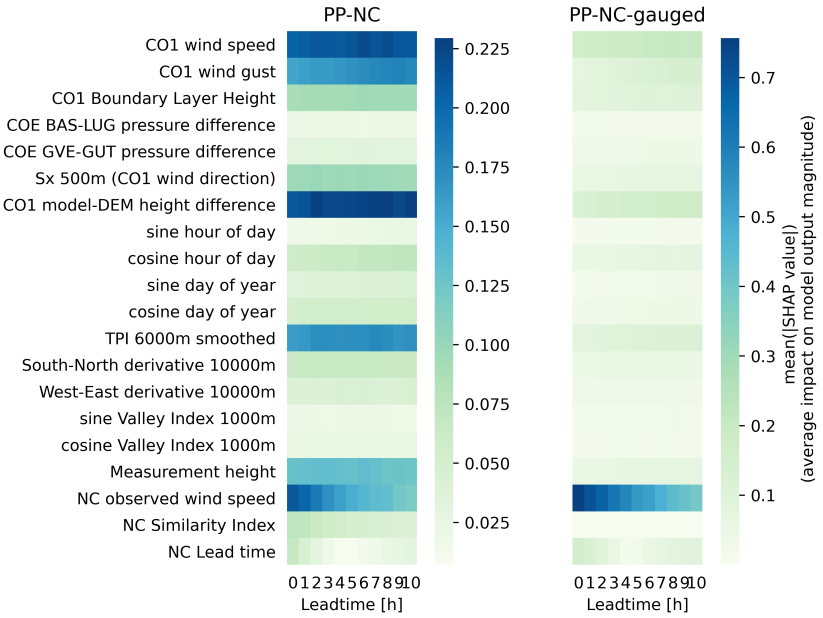

Fig. 4.10.: Heat map visualizations of the average impact on the model output magnitude for each predictor, for the PP-NC model at ungauged (left) and gauged (right) locations.

For all our explanations, the baseline value is 2.14 m/s. So if we were to predict a wind speed of 1 m/s, the sum of SHAP values from all predictors would be equal to -1.14 m/s. Finally, we recall that the explanations were conducted by considering the mean of the output CPD, so the uncertainty does not play a direct role.

To see how our model behaviour changes with the leadtime, we can aggregate SHAP values based on leadtimes from 0h to 10h, and compute the average for each predictor. This is shown in Fig. 4.10, for both the ungauged and gauged settings. Note that the pixel value scale is not the same for the two figures, as we wanted to highlight the variation rather than the magnitude itself. First, we note that the observed wind speed has a much larger influence at gauged stations, which confirms what we already stated before. That is, the model can use the Similarity Index as a weight for the observed value. In both cases we see a change in the importance of predictors as we go from 0h to 10h leadtime: the average impact of the observed wind speed descreases while it increases or stays constant for other predictors. Therefore, to some degree the model is able to weigh predictors based

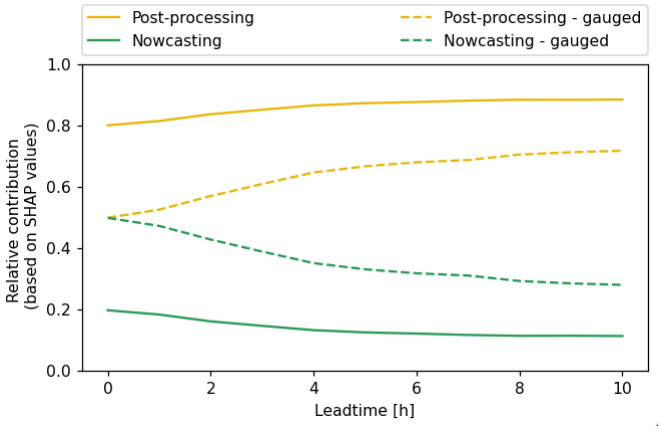

Fig. 4.11.: Average relative contribution of post-processing related predictors and nowcasting related predictors to the model output magnitude. PP-NC contributions at ungauged and gauged locations are shown with solid and dashed lines respectively. From the perspective of seamless prediction methods, this figure can be interpreted as an *average blending function* of the model.

on leadtime, but not sufficiently. This is particularly true for gauged locations. If we combine the post-processing related predictors and the nowcasting-related predictors in two separate groups, take the sum and normalize for the total, we can calculate their relative contribution to the model output as shown in Fig. 4.11. For the gauged locations, the relative contribution is roughly the same at 0h leadtime, when ideally we would expect the nowcasting component to be the main factor. On the other hand, the contribution at 10h leadtime is in line with what we would expect based on the average autocorrelation of wind speed for our stations (see A.1).

Model limitations

We now have enough information to formulate some hypotheses about why the model does not behave exactly as expected concerning the use of real-time information at gauged locations and for very short leadtimes (we focus on the analysis because it is where this misbehavior is more evident). So far, we determined that there are two sources of error: one is the overestimation of the uncertainty, the other is the fact that the observed wind speed does not receive enough importance. Let us focus on the former. We believe that there are two fundamental issues at play. One is regarding the parameterization of the Gamma distribution (see 3.12), which

makes it inherently more difficult (from the point of view of a neural network) to produce distributions with a large mean and small standard deviation. These are defined as

$$\mu = \frac{\alpha}{\beta}, \tag{4.1}$$

$$\sigma = \frac{\sqrt{\alpha}}{\beta}, \tag{4.2}$$

where α and β are the concentration and rate parameters respectively. From this relations we deduce that for a given mean value (the ratio of α and β remains constant), the magnitude of α and β increase fast as the standard deviation decreases. An example is shown in B.2 and B.3: if we were to predict a distribution centered around 15 m/s and a standard deviation of 0.43 m/s, concentration and rate parameters would need to be 1215 and 81 respectively. For a standard deviation of 3.87 that would be 15 and 1. Therefore, for a relatively small difference in terms of model output, the change in the distribution parameters is very large and may pose additional difficulty for the model in distinguishing between different situations. A possible solution that could be implemented to address this is to change the predictive distribution, e.g. to a normal distribution truncated at zero to avoid negative values. An attempt was made during this project, but for unknown reasons (probably a software issue with Tensorflow Probability) the training program failed during execution.

Another issue is related to the way we introduce empirical uncertainty. When we formulate the predictive distribution using the BMA, we *marginalized* the model parameters. That is, the predictive distribution is not directly dependend on the model parameters w but ultimately on the training data D. The consequence is that the composition of D heavily determines uncertainty in the model predictions. Predictions for situations that are very well represented (e.g. low wind) have very little empirical uncertainty (so they are almost identical to a non-Bayesian equivalent), whereas poorly represented examples (e.g. high wind speed) have large empirical uncertainty. While this approach is justified for most situations,

using the target value as a predictor in the case of analysis at gauged locations defies its underlying logic. For the empirical uncertainty associated with an observed wind speed in such case is by definition null (if we do not account for measurement uncertainty, but this is an entirely different issue) irrespective of how well it is represented in the training dataset. This is a straightforward argument for us, but the model is not aware of it and predicts with high empirical uncertainty even when this is not needed. In order to substantiate this hypothesis a model was trained to predict only the analysis at gauged locations, and it yielded similar results: a still relatively high CRPS of 0.21 and too large uncertainty for large wind speed. However, when the same model was trained without dropout, the CRPS dropped to 0.03. To overcome this problem, a promising approach would be to add constraints to the model, such that it is forced not to introduce empirical uncertainty under specific circumstances. This idea of adding knowledge-based constraints to machine learning models is becoming increasingly popular, and a strong case for its implementation in the atmospheric sciences is made in (Kashinath et al., 2021).

As we mentioned in the beginning of this section, a second source of error concerning the use of real time information is that the observed wind speed does not receive enough importance for very short leadtimes. We speculate that this behavior is related to gradient-based optimization. During each training step, the model parameters are updated such that more importance is given to those features that explain more variability in the training batch, irrespective of their actual "relevance" (this is something of which we, as humans, are aware but not a machine). Consequently, at short leadtimes, even though the real time information explains most of the variability, the other predictors still explain a part of it too and the model is tricked into relying on them instead of just using the real time information. Regardless of the true reason for this unwanted behavior, adding knowledge-based constraints would be an effective measure in this case too as explained in the above paragraph.

On the effect of the Similarity Index

Since more importance is given to wind speed observations at gauged locations, we already determined that the Similarity Index is able to act as a weight for the

observed wind speed. However, the distinction between ungauged and gauged location is a fairly obvious one. To actually look at the impact of the observed wind for different values of Similarity Index, we can look at SHAP values of the PP-NC model at ungauged locations. Specifically, it is useful to look at the SHAP values of observed wind speed as a function of the Similarity Index. This is shown in Fig. 4.12. The color indicates the magnitude of the input value of observed wind. The impact on the model output magnitude increases with the Similarity Index. We believe this result is particularly significant in that it proves that the Similarity Index is a useful metric and it is fit for its purpose. While certainly there are ways to improve its derivation, and perhaps define it more formally, these results provide a proof of concept.

Ways to explore potential improvements for the Similarity Index include: the use of different criteria for stratification instead of the weather type; an additional step that scales the Similarity Index based on the proportion of variances of a pair of stations, thus considering a "predictand" and a "predictor" station. Such procedure would effectively transform the Similarity Index into a metric that is analogous to the β coefficient of a simple linear regression.

4.3 Model explainability

In this section we explore some of the results obtained by using SHAP for model explainability, from a more general standpoint than just from the point of view of nowcasting. We will look at how our predictors affect predictions and look for interesting co-dependencies, while also discussing some counter-intuitive results. Then, we will present examples of how SHAP values may be used to interpret single predictions. The goal of this section is to provide examples of how model explainability can be useful to validate machine learning models and build thrust in their results. We should always keep in mind that these explanations represent how the model arrives to its predictions without any knowledge of the process of interest, but based exclusively on statistical relationships that it integrated in a non-linear fashion. In this context, the main point of model explainability is not one of knowledge *discovery*, but rather of *confirmation*.

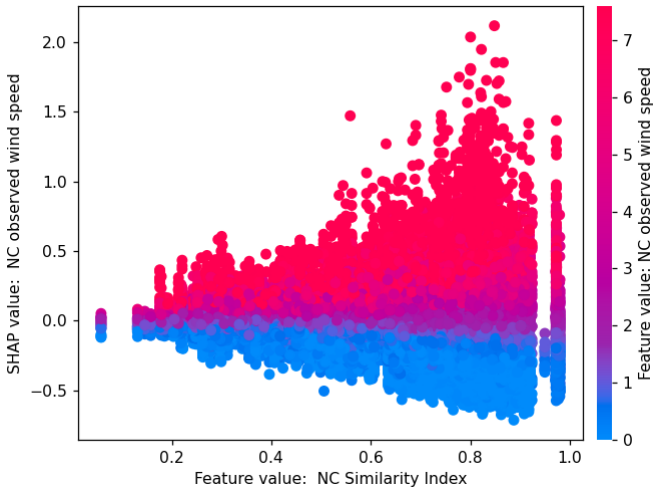

Fig. 4.12.: Scatter plot representing SHAP values for the observed wind speed as a function of the Similarity Index. Each dot represents a sample prediction, with roughly 40'000 predictions being displayed, and it is colored based on the actual input value (not normalized) of the observed wind speed. The impact on the model output magnitude increases with the Similarity Index. The gaps for high values occurs simply because very few stations in our dataset are sufficiently *similar* to each other.

4.3.1 Summary visualization

A convenient way to summarize the results of model explanations with SHAP is by using the so-called "beeswarm" plot. Like a box-plot it is able to convey information about the data distribution, but it does so by displaying individual samples as points, and adjusting their position in order to reduce overlaps. These points can than be color-coded based on the feature's input value. This visualization is presented in Fig. 4.13. Approximately 40'000 sample predictions are displayed in the figure, where each prediction corresponds to a set of 20 points (one for each feature) placed along the horizontal axis. Features are sorted by decreasing importance from top to bottom. We will only look at the most relevant ones. Without surprise, the a relevant feature of the model is the COSMO-1 wind speed. Small values of COSMO-1 wind speed have a relatively low, negatively oriented impact on the model output, whereas positively oriented impact for large values can be much higher. This is due to the fact that the baseline of 2.14 m/s is also relatively low compared to the range of values in the sample. Among our topographical descriptors, the model-DEM height difference is the most important feature. Low feature values indicate when

the DEM height is higher than the model height, and we observe a large positive impact on the model output in such situations. A similar result is shown for the TPI at 6000m scale (although in this case the values are to be interpreted with the opposite sign). Both features seem to correct systematic and conditional biases related to the exposure of a location in relation to a sub-grid scale (narrow valleys and crests) and to 6000m scale (wide valleys and ridges). The results for the measurement height are a good indication that diverse conditions can be included successfully in the training dataset, provided that the meta-information is reliable. The Sx proved to be an important topographical descriptor, being the only one to greatly impact model output both positively and negatively. We note that since this descriptor if flow-dependent, its usefulness strongly depends on the accuracy of the NWP prediction of wind direction. Therefore, we can expect that its importance for nowcasting applications is generally higher than for post-processing at medium range predictions. Interestingly, there are a few examples of strong COSMO-1 wind gusts having a negative effect on the model output. This is an unexpected behavior, probably resulting from collinearities between COSMO-1 wind gusts and other predictors (such as COSMO-1 wind speed or observed wind speed). The same visualization for gauged locations is shown in B.7

4.3.2 Feature dependence

By combining information about multiple features, we are able to get insights into their relationship within the model. We will look at two interesting examples. First, let us consider the effect of the Sx topographical descriptor and the COSMO-1 wind speed, shown in Fig. 4.14 on the left. On the vertical axis is the impact of the Sx on the model output, on the horizontal axis its input values and each prediction represented by a single dot is colored based on the value of COSMO-1 wind speed. We observe that for low values of COSMO-1 wind speed samples are distributed almost horizontally, indicating a weak impact of the Sx on the model output, whereas in the case of high values the effect of the Sx is greater. We can deduce that the Sx has a dampening and amplifying effect that depends on COSMO-1 wind speed. Upwind, exposed slopes increase wind speed proportionally to its magnitude, and to opposite happens for downwind, sheltered slopes.

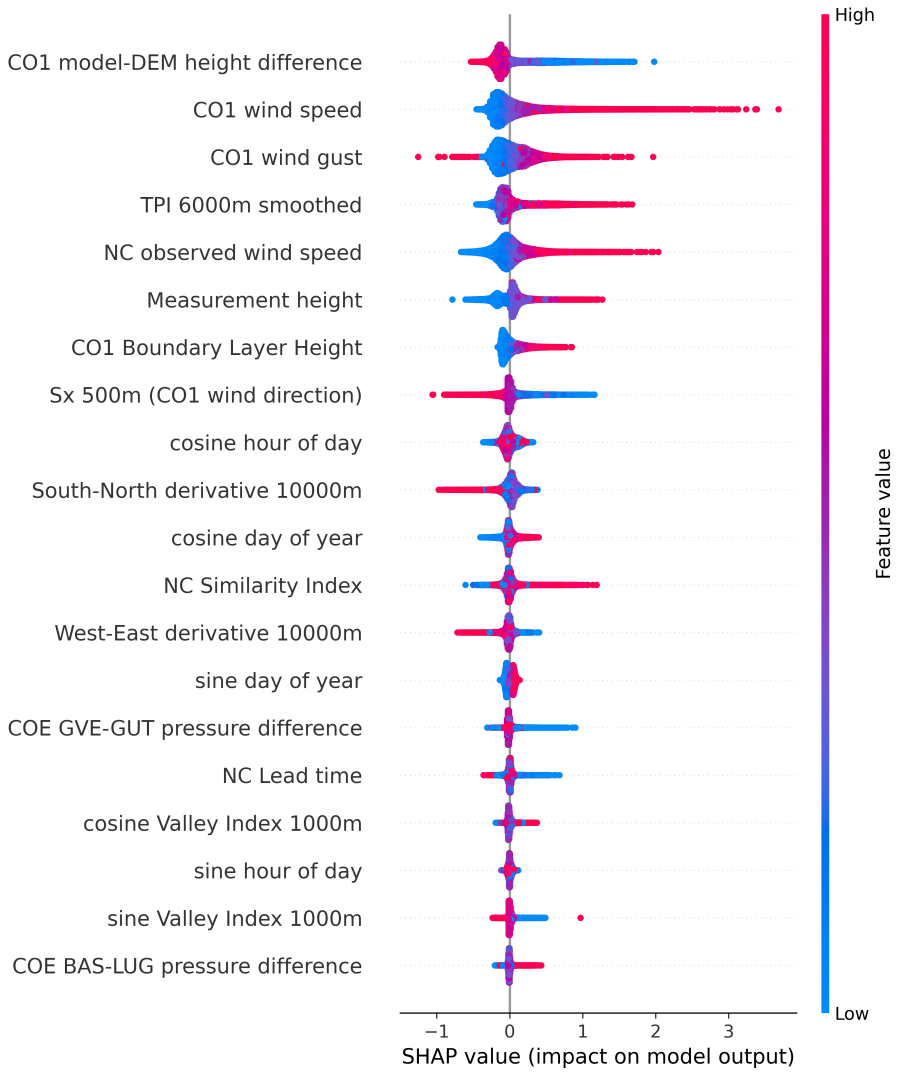

Fig. 4.13.: Beeswarm plot representing SHAP values of the PP-NC model at ungauged locations for each model predictor, color-coded based on feature's input values. One sample prediction corresponds to a set of 20 points, one for each feature, placed along the horizontal axis.

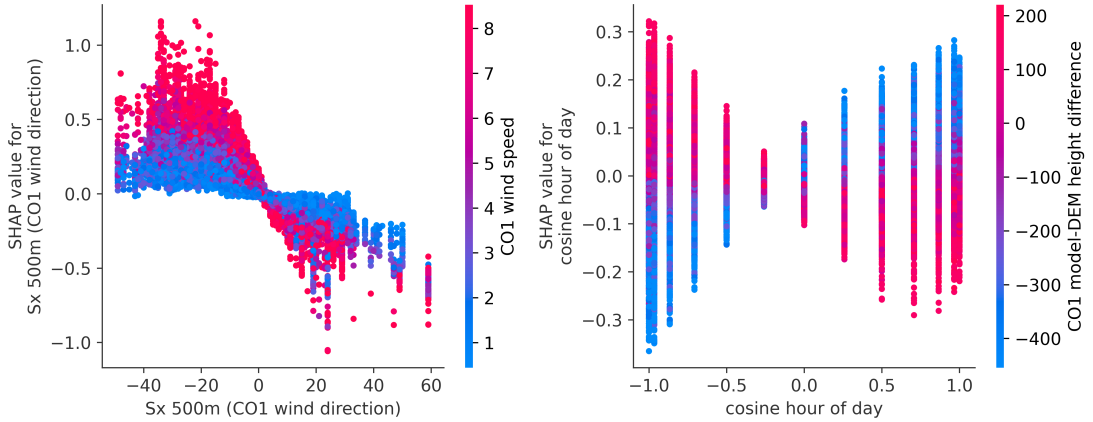

Fig. 4.14.: Left: plot of the impact of the Sx topographical descriptor as a function of its input value, color-coded based on the input value of COSMO-1 wind speed. Right: scatter plot of the impact of the cosine component of the hour of the day as a function of its input value, color-coded based on the input value of the height difference between COSMO-1 model topography and the high resolution DEM.

Another interesting example of feature dependence is regarding the diurnal cycle. Well known surface wind circulation patterns are particularly important in the alpine area. We can therefore expect that the impact of the hour of the day is not equal everywhere, but is strongly dependent on the geomorphological setting of a location. Thanks to SHAP values we can verify this, as shown in Fig. 4.14 on the right. The figure shows the impact of the cosine component of the hour of the day (which is in phase with the day-night cycle, with -1 occurring at midnight and 1 occurring at noon) as a function of its input value and color-coded with respect to the value of the height difference between COSMO-1 topography and an high resolution DEM. We observe that in valley bottoms (red dots) there is a positive impact on wind speed during the day and a negative impact during the night, whereas the opposite is true for mountain crests (blue dots). This confirms our expectations based on prior knowledge, and it is a good indication that the model integrated relevant relationships in an automated way.

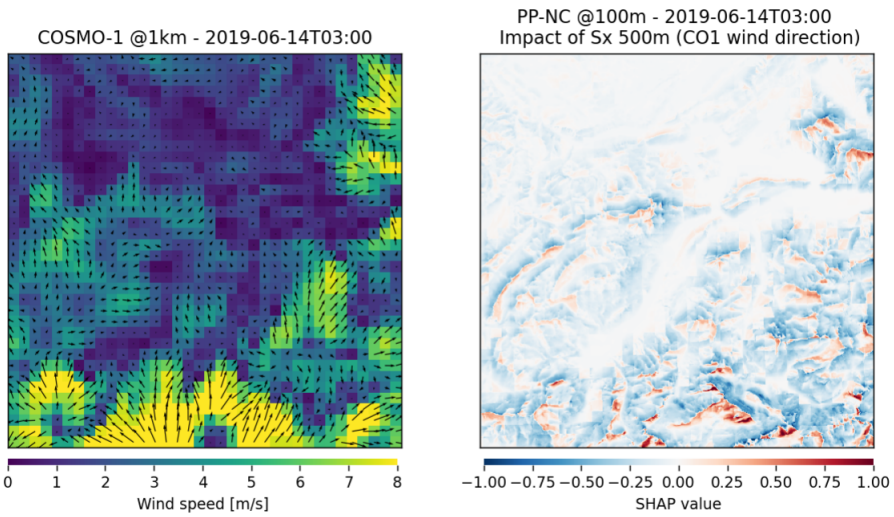

Fig. 4.15.: On the left, the COSMO-1 wind speed magnitude field displayed with wind vectors. On the right, the impact of the Sx topographical descriptor on the model output. The spatial distribution of SHAP values for Sx closely follows the magnitude and direction of COSMO-1 wind field.

4.3.3 Spatial SHAP analysis

In the previous sections we have only considered SHAP values for a set of random samples from our test dataset. While this allows us to make general interpretations about the model, they are somewhat limiting when we are interested in the spatial structure of SHAP values and whether it looks realistic or not. By computing SHAP values over a spatial domain for specific timesteps, we are able to interpret single events. An example of this is presented in Fig. 4.15 for the impact of Sx, with the wind speed vectors of COSMO-1 displayed to the left. As expected, the Sx has a positive impact on upwind slopes and a negative impact on downwind slopes. Moreover, the magnitude of the impact appears proportional to the input wind speed from COSMO-1.

Figure 4.16 shows another example of a prediction in our study domain, along with the impact of some of the predictors. The upper right plot represents the impact of the real-time observations of wind speed. Intuitively, this is a way to visualize the intrinsic interpolation of the PP-NC model. For an additional example see B.8.

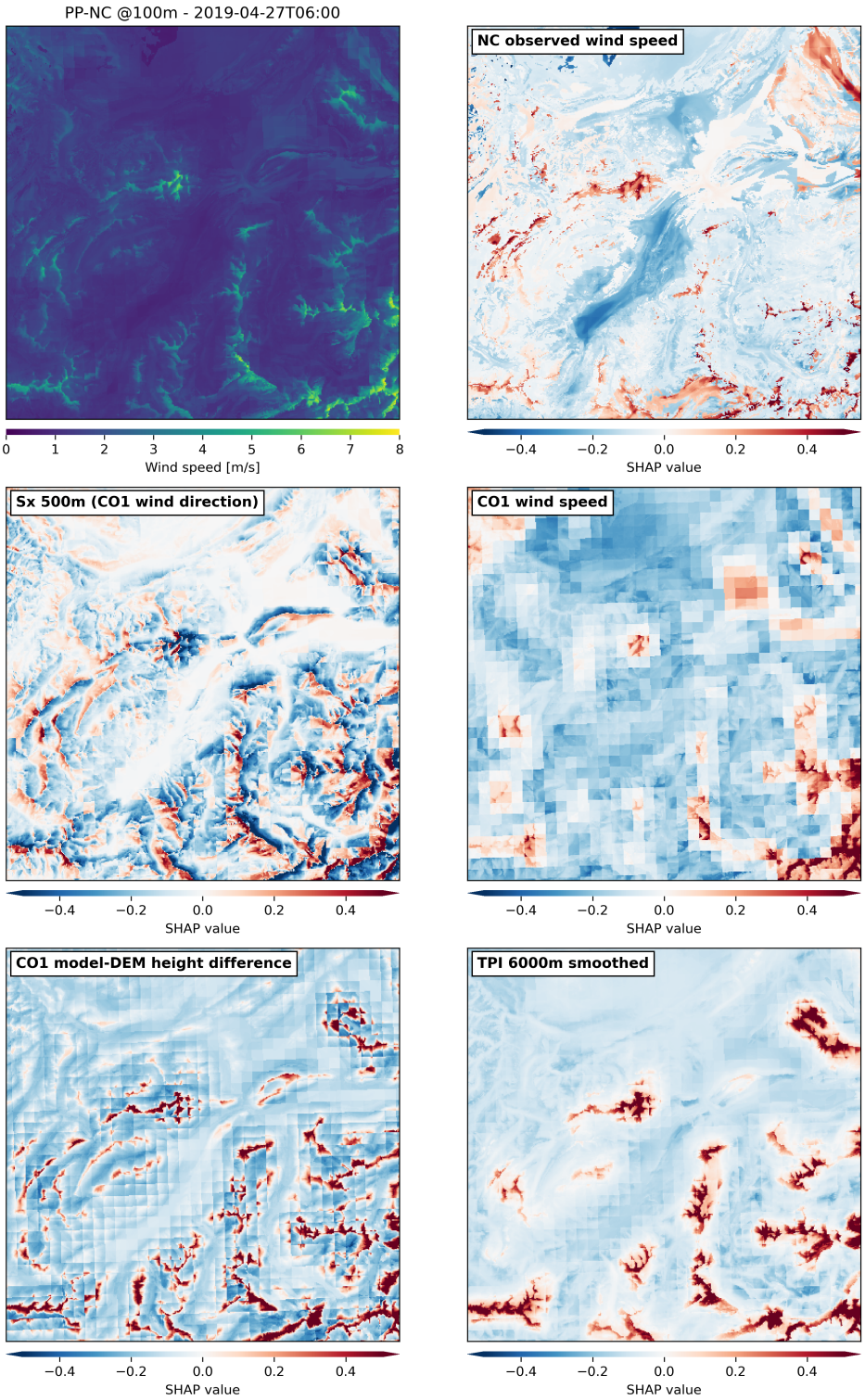

Fig. 4.16.: A single prediction on the study domain with the impacts of some of the predictors.

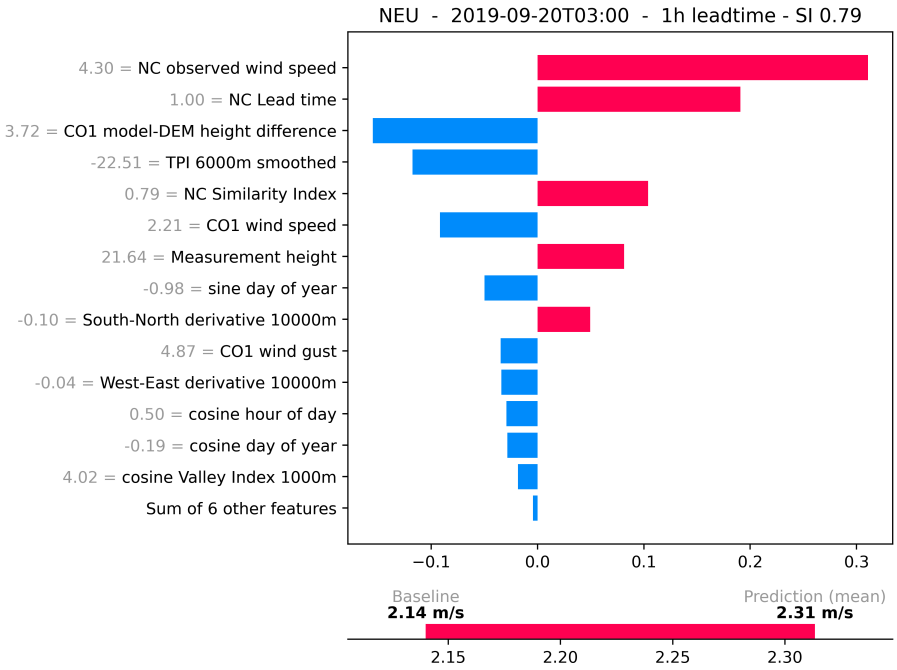

Fig. 4.17.: Model explanation of a single prediction at Neuchâtel (NEU). The horizontal axis represents the impact of a predictor on the final output of the model, in blue and red for negative and positive impact respectively. The bottom plot shows the final prediction compared to the baseline, where the difference between the two is the sum of all contributions in the above plot.

4.3.4 Explaining individual predictions

Another interesting use of SHAP values is to represent explanations of single predictions, and directly observe how each predictor impacted the model output for each specific case. An example is presented in Fig. 4.17. In this situation, we see a good example of how despite the large wind speed input of COSMO-1 the model predicts a relatively low value, due in great part to the influence of the Sx, indicating that the location in a sheltered position relative to the direction of the wind. Another example is shown in B.9.

5 Conclusions

Following recent efforts in this domain, ANN-based post-processing of surface wind speed in complex topography was further investigated. This work helped in consolidating and further developing a new framework used at MeteoSwiss to facilitate research in this field. Several contributions were made to the code base, including bug fixes and additions of new tools and resources.

A new metric called Similarity Index was developed. The Similarity Index allows to estimate the correlation of wind speed between two locations, based on their geographical position and geomorphological settings. For its derivation, we trained an ANN that also included a weather type classification as predictor, as a way to stratify weather conditions and make a step towards flow dependency. The model was able to estimate correlations with a mean absolute error of 0.092 on an independent test dataset. The Similarity Index was discussed, and we highlighted the importance of topographical predictors as well as the weather situation in determining the spatial correlation structure in complex topography. Additionally, we showed that the Similarity Index conveys useful information about the representativity of the measuring network.

A new methodology to include real time information in ANN-based post-processing of surface wind speed was developed. This approach makes use of the Similarity Index to find the most representative wind speed observation at any given location and at any given time, and determine the impact of said observation on the final output of the model. In practice, we successfully introduced real time observations of wind speed in an optimized way that mimics geostatistical methods of interpolation such as regression Kriging, but without sacrificing performance since the Similarity Index can be computed offline. The model that included this nowcasting component has shown a significant improvement in performance compared to the baseline and a simple post-processing model, with a sharp decrease of CRPS at gauged locations but also a noticeable decrease at locations further away from gauged locations. This last aspect is particularly important in that it supports

the generalising capabilities of our new methodology. The way the model uses real-time information was discussed, indicating a correct overall behavior but also a few shortcomings, for which potential solutions were suggested.

The state of the art explainability technique SHAP was used to provide insights into the *black box* model. We have shown how SHAP values may be used in several ways to gain trust in ANN post-processing models, by providing direct examples for our application.

5.1 Outlook

The Similarity Index has proved to be a useful metric in our case. Conceptually, the most important aspect is that we are able to mimic a statistically optimized spatial interpolation without the need to compute semi-variograms at every timestep. Building on top of this core idea, we believe there are still ways to improve its derivation or even reformulate it. For instance, it would be interesting to develop alternative solutions for stratification based on weather conditions, making use of additional meteorological parameters. The increasing amount of observational data coming from different sources (and perhaps including crowd-sourced data) calls for specific algorithms to ensure that only high quality measurements are included in the training dataset. Additionally, collecting accurate meta-information about weather stations may allow to make better use of measurements coming from a diverse set of conditions. We believe further improvements in this field will increasingly be focused on finding ways to constrain machine learning models with our prior knowledge, as suggested in section 4.2.2.

Data and methods

A

Tab. A.1.: Number of stations in the measuring network operated by each organization.

Name of the organisation	Number of stations
ARPA Lombardia	51
Autonome Provinz Bozen - Südtirol	70
Botanisches Institut der Universität Basel	1
Bundesamt für Umwelt	8
Deutscher Wetterdienst	42
Eidg. Forschungsanstalt WSL	17
Eidg. Institut für Schnee- und Lawinenforschung	180
Kachelmannwetter GmbH	14
Kanton Aargau	3
Kanton Graubünden	8
Kanton Thurgau	20
Kanton Wallis; Dienststelle für Umweltschutz	5
Kanton Wallis; Dienststelle für Wald und Lands...	4
Lufthygieneamt beider Basel	3
MeteoGroup Schweiz AG	54
MeteoSchweiz	150
Ostluft	9
Regione Autonoma Valle d'Aosta	26
Repubblica e Cantone Ticino	5
République et canton de Neuchâtel	5
Schweizer Armee - Luftwaffe	7
Schweizerischer Nationalpark	1
Swiss Permafrost Monitoring Network	6
Windguru / Martin Schuler	1
Zentralanstalt für Meteorologie und Geodynamik	46
inNET Monitoring AG	3
Total	739

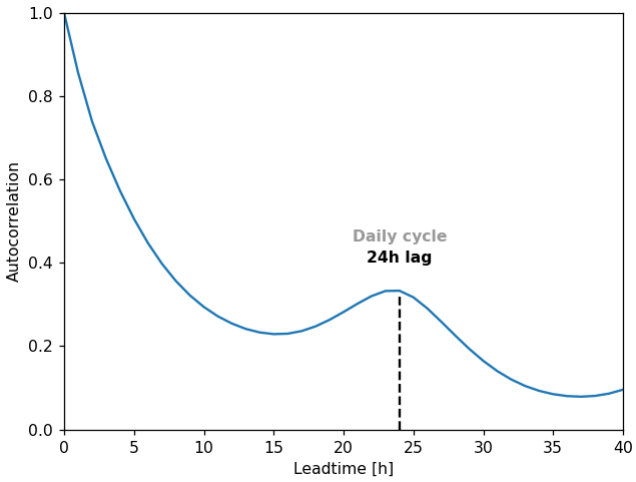

Fig. A.1.: Mean autocorrelation of wind speed for all selected stations from our dataset.

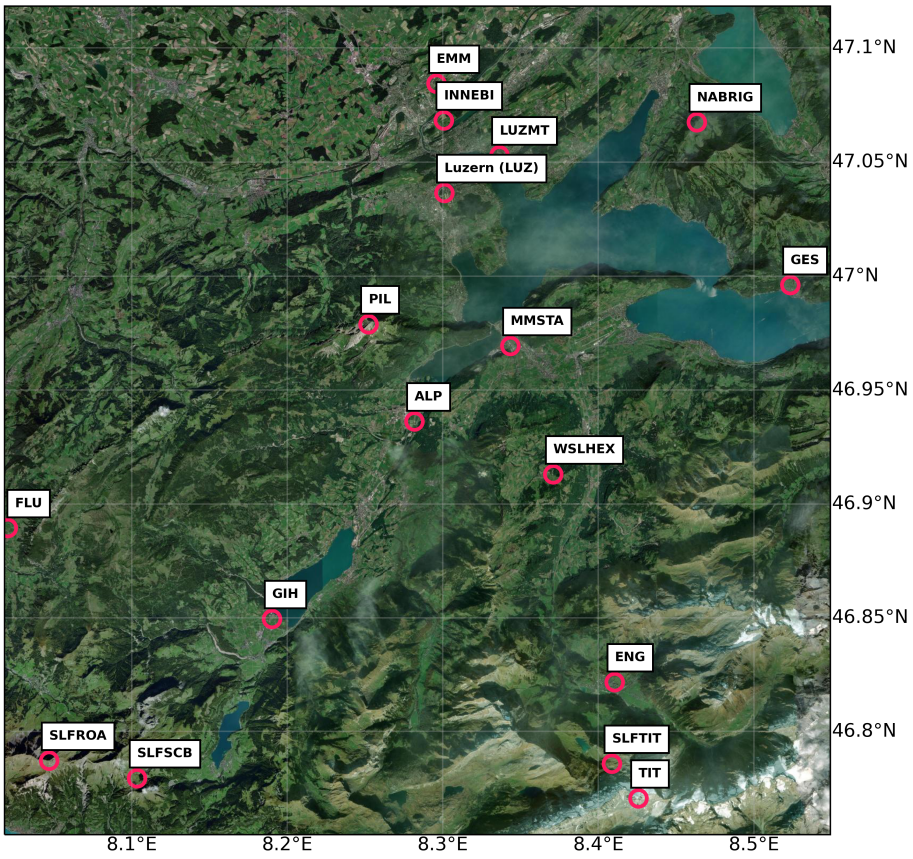

Fig. A.2.: Spatial domain used to present results of Similarity Index and PP-NC model.

Results

B

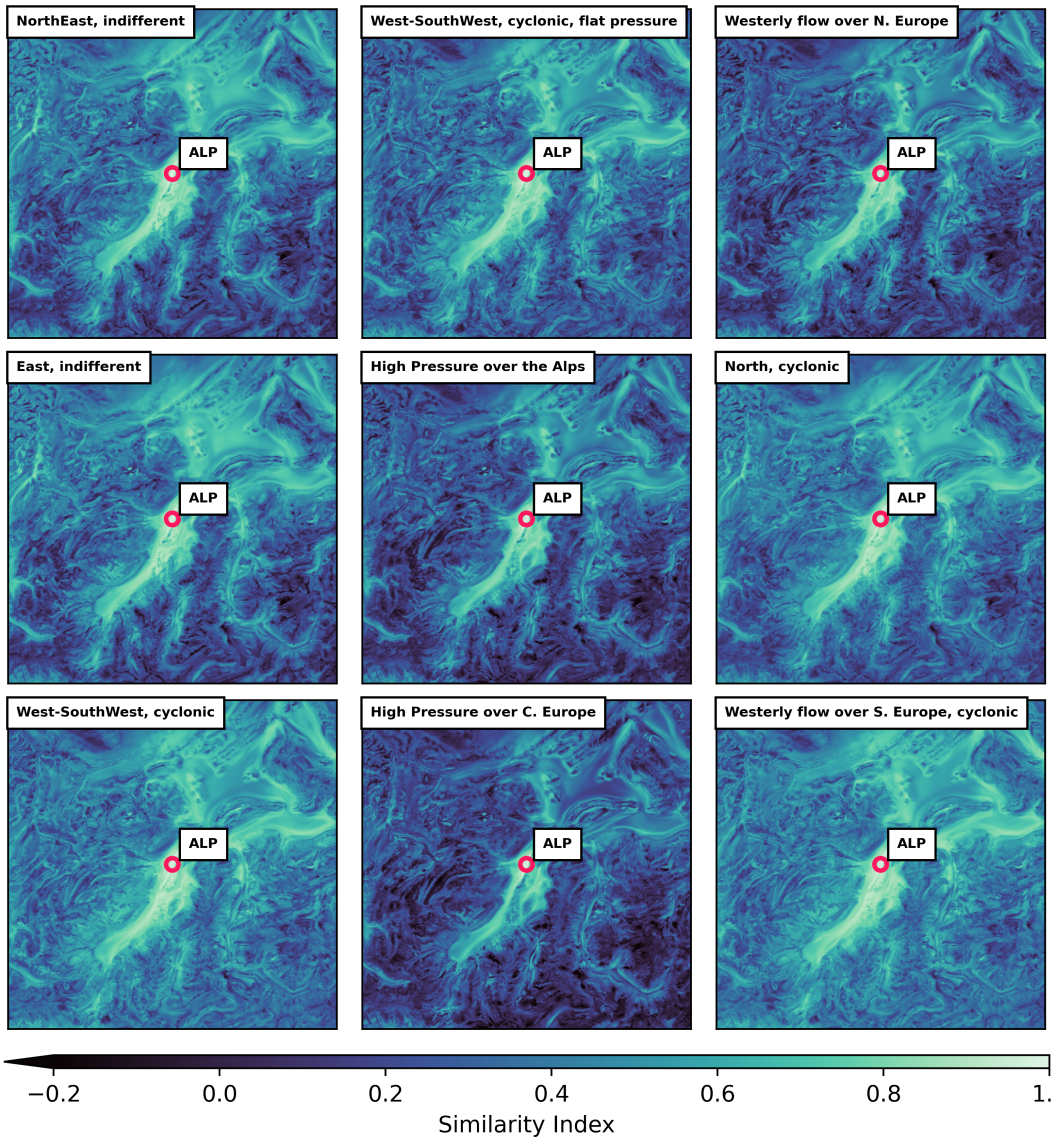

Fig. B.1.: Similarity Index with respect to the weather station located in Alpnach for all CAP9 weather classification codes.

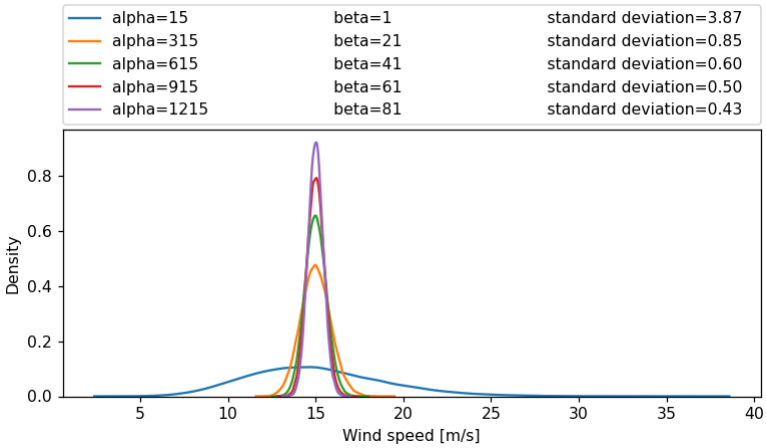

Fig. B.2.: Experiment showing sample Gamma distributions centered around 15 m/s with different parameterizations, resulting in increasingly sharp distributions.

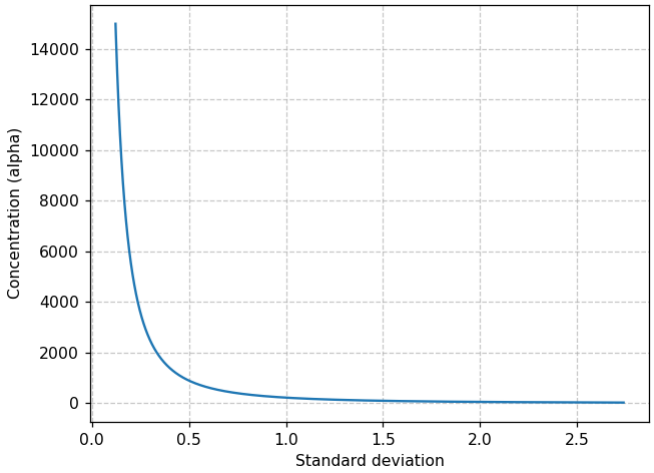

Fig. B.3.: Concentration parameter of a Gamma distribution centered around 15 m/s, as a function of the standard deviation. The concentration parameter displays an asymptotic behavior as the distribution approaches a standard deviation of zero.

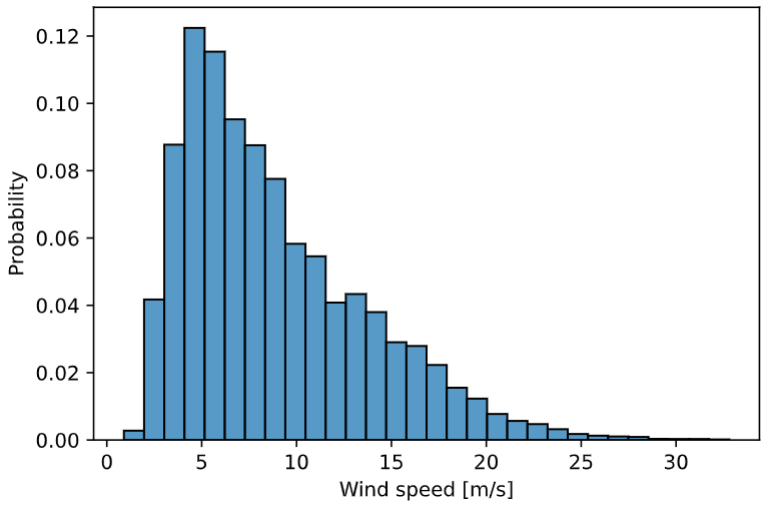

Fig. B.4.: Distribution of observed wind speed for events where all ensemble members of the PP-NC model prediction were smaller than the observed value.

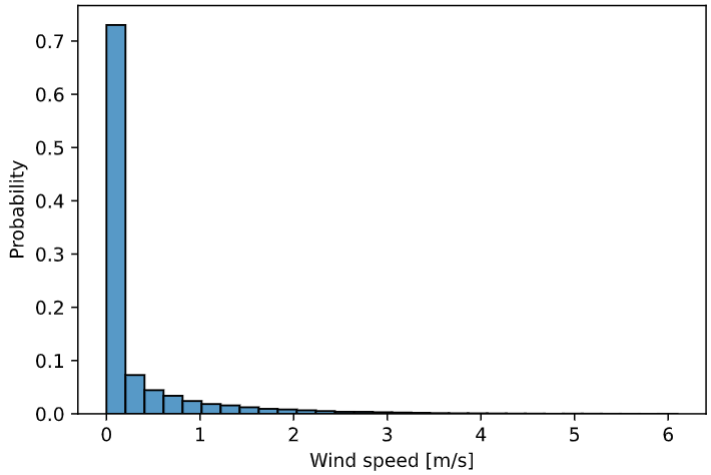

Fig. B.5.: Distribution of observed wind speed for events where all ensemble members of the PP-NC model prediction were larger than the observed value.

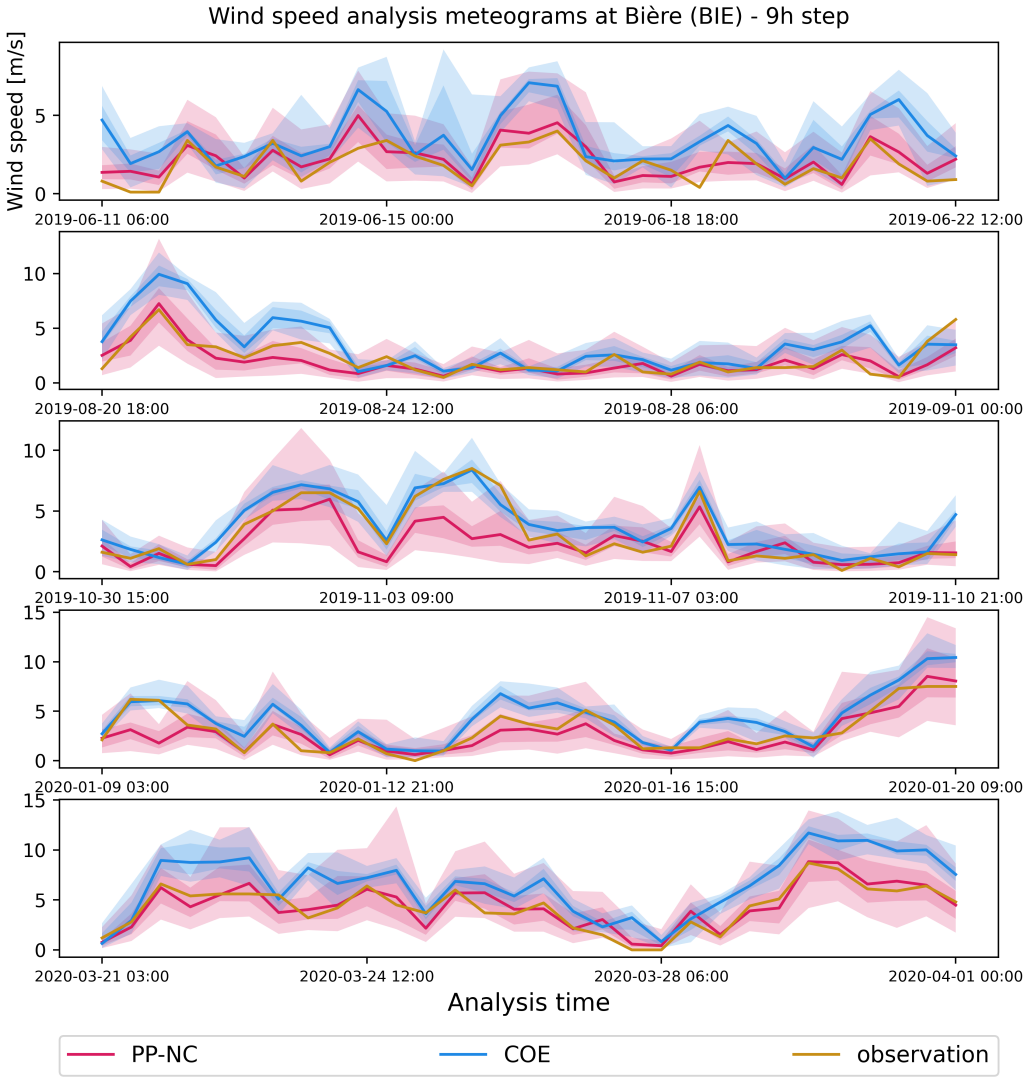

Fig. B.6.: Examples of analysis meteograms for wind speed at Bière (BIE), without using the measurements at the station itself.

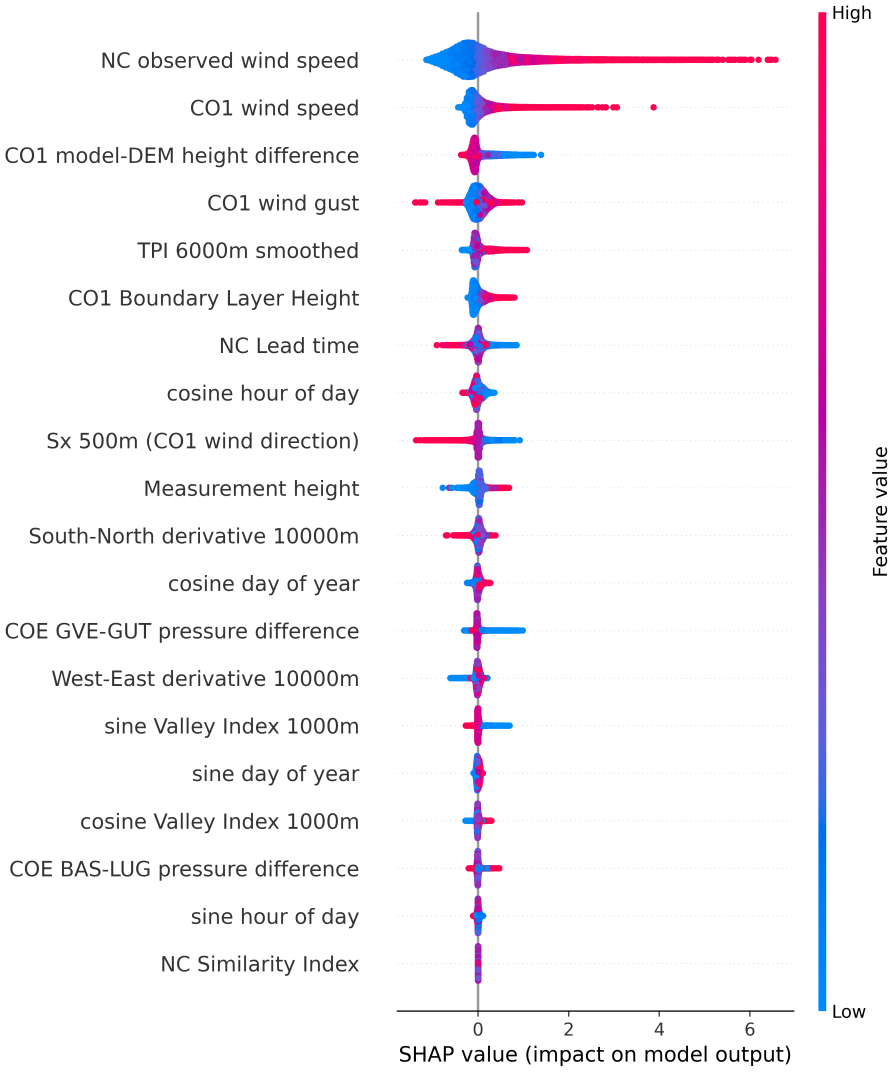

Fig. B.7.: Beeswarm plot representing SHAP values of the PP-NC model at gauged locations for each model predictor, color-coded based on feature's input values. One sample prediction corresponds to a set of 20 points, one for each feature, placed along the horizontal axis.

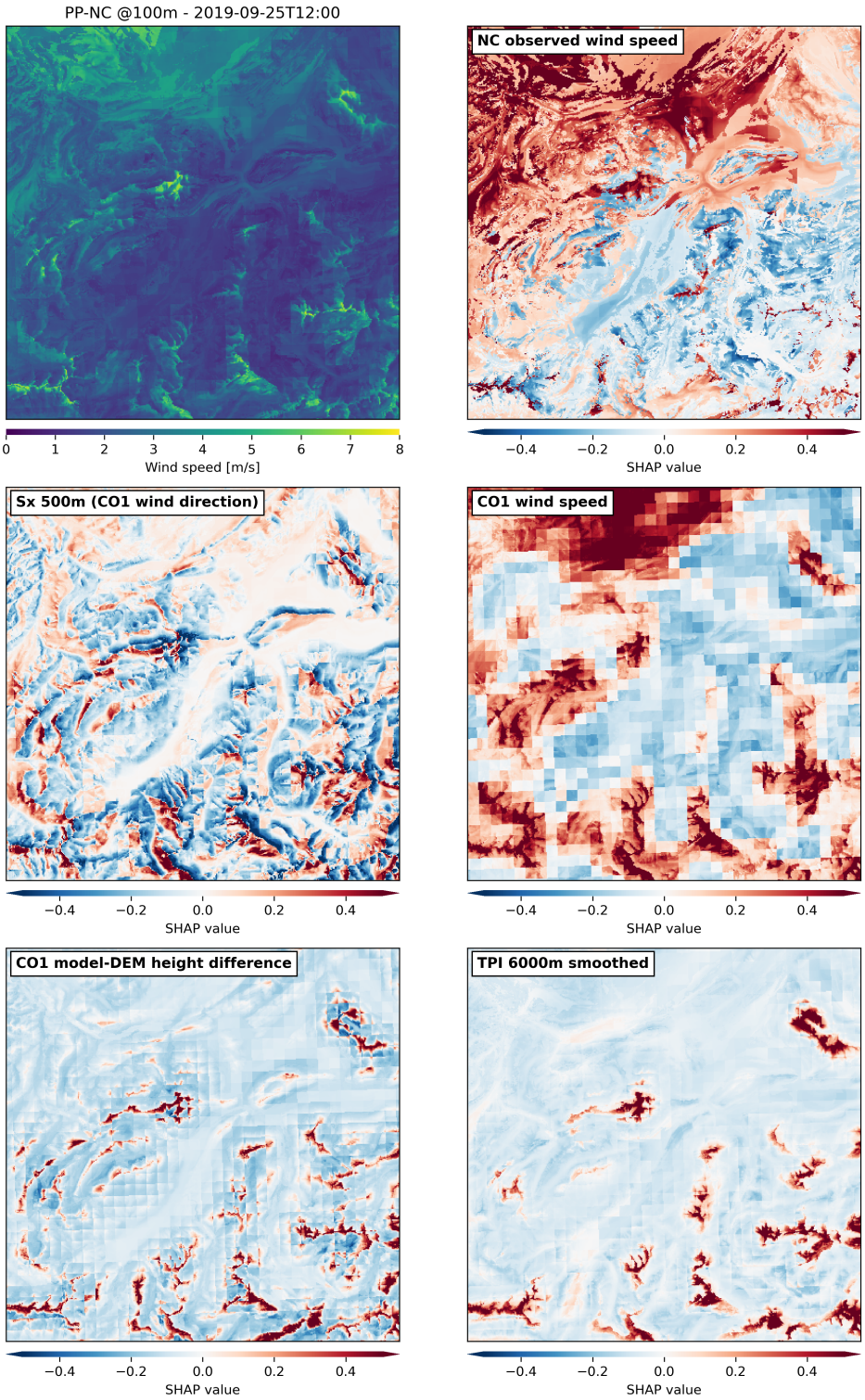

Fig. B.8.: A single prediction on the study domain with the impacts of some of the predictors.

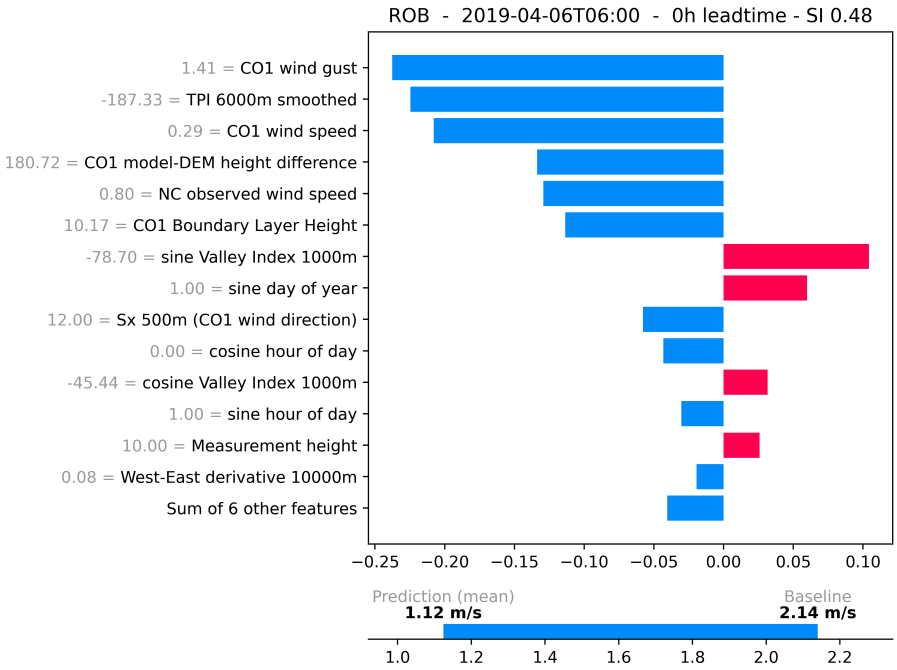

Fig. B.9.: Model explanation of a single prediction at Poschiavo / Robbia (ROB). The horizontal axis represents the impact of a predictor on the final output of the model, in blue and red for negative and positive impact respectively. The bottom plot shows the final prediction compared to the baseline, where the difference between the two is the sum of all contributions in the above plot.

Bibliography

Bauer, P., A. Thorpe, and G. Brunet (2015). „The quiet revolution of numerical weather prediction“. en. In: *Nature* 525.7567, pp. 47–55. DOI: 10.1038/nature14956.

Bremnes, J. B. (2020). „Ensemble Postprocessing Using Quantile Function Regression Based on Neural Networks and Bernstein Polynomials“. EN. In: *Monthly Weather Review* 148.1, pp. 403–414. DOI: 10.1175/MWR-D-19-0227.1.

Buzzi, M., M. Guidicelli, and M. A. Liniger (2019). „Nowcasting wind using machine learning from the stations to the grid“. In: *Agencia Estatal de Meteorología*.

Carta, J. A., P. Ramírez, and S. Velázquez (2009). „A review of wind speed probability distributions used in wind energy analysis: Case studies in the Canary Islands“. en. In: *Renewable and Sustainable Energy Reviews* 13.5, pp. 933–955. DOI: 10.1016/j.rser.2008.05.005.

Cervone, G., L. Clemente-Harding, S. Alessandrini, and L. Delle Monache (2017). „Short-term photovoltaic power forecasting using Artificial Neural Networks and an Analog Ensemble“. en. In: *Renewable Energy* 108, pp. 274–286. DOI: 10.1016/j.renene.2017.02.052.

Chapman, W. E., A. C. Subramanian, L. D. Monache, S. P. Xie, and F. M. Ralph (2019). „Improving Atmospheric River Forecasts With Machine Learning“. en. In: *Geophysical Research Letters* 46.17-18, pp. 10627–10635. DOI: https://doi.org/10.1029/2019GL083662.

Chen, H., J. D. Janizek, S. Lundberg, and S.-I. Lee (2020). „True to the Model or True to the Data?“ In: *arXiv:2006.16234 [cs, stat]*.

Chollet, F. (2015). *Keras*.

Chollet, F. (2017). *Deep Learning with Python*. 1st. USA: Manning Publications Co.

Dai, Y. (2020). „Post-processing cloud cover forecasts using Generative Adversarial Networks“. MA thesis. ETH Zürich.

Dürr, O., B. Sick, and E. Murina (2020). *Probabilistic Deep Learning: With Python, Keras and TensorFlow Probability*. Manning Publications.

Fundel, V. J., N. Fleischhut, S. M. Herzog, M. Göber, and R. Hagedorn (2019). „Promoting the use of probabilistic weather forecasts through a dialogue between scientists, developers and end-users“. en. In: *Quarterly Journal of the Royal Meteorological Society* 145.S1, pp. 210–231. DOI: https://doi.org/10.1002/qj.3482.

Gal, Y. and Z. Ghahramani (2016). „Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning“. In: *arXiv:1506.02142 [cs, stat]*.

Gneiting, T., F. Balabdaoui, and A. E. Raftery (2007). „Probabilistic forecasts, calibration and sharpness“. en. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 69.2, pp. 243–268. DOI: https://doi.org/10.1111/j.1467-9868.2007.00587.x.

Gneiting, T. and M. Katzfuss (2014). „Probabilistic Forecasting“. In: *Annual Review of Statistics and Its Application* 1.1, pp. 125–151. DOI: 10.1146/annurev-statistics-062713-085831.

Gneiting, T. and A. E. Raftery (2007). „Strictly Proper Scoring Rules, Prediction, and Estimation“. In: *Journal of the American Statistical Association* 102.477, pp. 359–378. DOI: 10.1198/016214506000001437.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep Learning*. MIT Press.

Grönquist, P., C. Yao, T. Ben-Nun, et al. (2020). „Deep Learning for Post-Processing Ensemble Weather Forecasts“. In: *arXiv:2005.08748 [physics, stat]*.

Hamill, T. M. (01 Mar. 2001). „Interpretation of Rank Histograms for Verifying Ensemble Forecasts“. In: *Monthly Weather Review* 129.3, pp. 550–560. DOI: 10.1175/1520-0493(2001)129<0550:IORHFV>2.0.CO;2.

Höhlein, K., M. Kern, T. Hewson, and R. Westermann (2020). „A comparative study of convolutional neural network models for wind field downscaling“. en. In: *Meteorological Applications* 27.6, e1961. DOI: https://doi.org/10.1002/met.1961.

Joslyn, S. and J. LeClerc (2013). „Decisions With Uncertainty: The Glass Half Full“. en. In: *Current Directions in Psychological Science* 22.4, pp. 308–315. DOI: 10.1177/0963721413481473.

Jospin, L. V., W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun (2020). „Hands-on Bayesian Neural Networks – a Tutorial for Deep Learning Users“. In: *arXiv:2007.06823 [cs, stat]*.

Jung, J. and R. P. Broadwater (2014). „Current status and future advances for wind speed and power forecasting“. en. In: *Renewable and Sustainable Energy Reviews* 31, pp. 762–777. DOI: 10.1016/j.rser.2013.12.054.

Kashinath, K., M. Mustafa, A. Albert, et al. (2021). „Physics-informed machine learning: case studies for weather and climate modelling“. In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 379.2194, p. 20200093. DOI: 10.1098/rsta.2020.0093.

Kuikka, I. (2009). *Wind Nowcasting: Optimizing Runway in Use*. Tech. rep. Helsinki University of Technology, Systems Analysis Laboratory: Espoo.

LeClerc, J. and S. Joslyn (2015). „The Cry Wolf Effect and Weather-Related Decision Making“. en. In: *Risk Analysis* 35.3, pp. 385–395. DOI: https://doi.org/10.1111/risa.12336.

Lehning, M. and C. Fierz (2008). „Assessment of snow transport in avalanche terrain“. en. In: *Cold Regions Science and Technology*. International Snow Science Workshop (ISSW) 2006 51.2, pp. 240–252. DOI: 10.1016/j.coldregions.2007.05.012.

Lewis, H. W., S. D. Mobbs, and M. Lehning (2008). „Observations of cross-ridge flows across steep terrain“. en. In: *Quarterly Journal of the Royal Meteorological Society* 134.633, pp. 801–816. DOI: https://doi.org/10.1002/qj.259.

Li, J. and A. D. Heap (2011). „A review of comparative studies of spatial interpolation methods in environmental sciences: Performance and impact factors“. en. In: *Ecological Informatics* 6.3, pp. 228–241. DOI: 10.1016/j.ecoinf.2010.12.003.

Lundberg, S. and S.-I. Lee (2017). „A Unified Approach to Interpreting Model Predictions“. In: *arXiv:1705.07874 [cs, stat]*.

Matheson, J. E. and R. L. Winkler (1976). „Scoring Rules for Continuous Probability Distributions“. In: *Management Science* 22.10, pp. 1087–1096. DOI: 10.1287/mnsc.22.10.1087.

Nicolis, C. (2007). „Dynamics of Model Error: The Role of the Boundary Conditions“. EN. In: *Journal of Atmospheric Sciences* 64.1, pp. 204–215. DOI: 10.1175/JAS3806.1.

Nicolis, C., R. A. P. Perdigao, and S. Vannitsem (2009). „Dynamics of Prediction Errors under the Combined Effect of Initial Condition and Model Errors“. EN. In: *Journal of Atmospheric Sciences* 66.3, pp. 766–778. DOI: 10.1175/2008JAS2781.1.

Papazek, P., I. Schicker, C. Plant, A. Kann, and Y. Wang (2020). „Feature selection, ensemble learning, and artificial neural Networks for Short-Range Wind Speed Forecasts“. In: *Meteorologische Zeitschrift*. DOI: 10.1127/metz/2020/1005.

Pindado, S., J. Cubas, and F. Sorribes-Palmer (2014). „The Cup Anemometer, a Fundamental Meteorological Instrument for the Wind Energy Industry. Research at the IDR/UPM Institute“. en. In: *Sensors* 14.11, pp. 21418–21452. DOI: 10.3390/s141121418.

Prechelt, L. (1998). „Early Stopping - But When?“ en. In: *Neural Networks: Tricks of the Trade*. Ed. by G. B. Orr and K.-R. Müller. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, pp. 55–69. DOI: 10.1007/3-540-49430-8_3.

Rasp, S. and S. Lerch (2018). „Neural Networks for Postprocessing Ensemble Weather Forecasts“. EN. In: *Monthly Weather Review* 146.11, pp. 3885–3900. DOI: 10.1175/MWR-D-18-0187.1.

Reinhardt, K. and C. Samimi (2018). „Comparison of different wind data interpolation methods for a region with complex terrain in Central Asia“. en. In: *Climate Dynamics* 51.9, pp. 3635–3652. DOI: 10.1007/s00382-018-4101-y.

Ribeiro, M. T., S. Singh, and C. Guestrin (2016). „"Why Should I Trust You?": Explaining the Predictions of Any Classifier“. In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pp. 1135–1144.

Ruder, S. (2017). „An overview of gradient descent optimization algorithms“. In: *arXiv:1609.04747 [cs]*.

Sättele, M., M. Bründl, and D. Straub (2016). „Quantifying the effectiveness of early warning systems for natural hazards“. English. In: *Natural Hazards and Earth System Sciences* 16.1, pp. 149–166. DOI: https://doi.org/10.5194/nhess-16-149-2016.

Schär, M. (2019). „Downscaling of gridded wind fields using Deep Learning methods“. MA thesis. Ècole Polytechnique Fédérale de Lausanne.

Scheuerer, M. and D. Möller (2015). „Probabilistic wind speed forecasting on a grid based on ensemble model output statistics“. EN. In: *Annals of Applied Statistics* 9.3, pp. 1328–1349. DOI: 10.1214/15-AOAS843.

Shrikumar, A., P. Greenside, and A. Kundaje (2019). „Learning Important Features Through Propagating Activation Differences“. In: *arXiv:1704.02685 [cs]*.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2014). „Dropout: a simple way to prevent neural networks from overfitting“. In: *The Journal of Machine Learning Research* 15.1, pp. 1929–1958.

Usaola, J., O. Ravelo, G. González, et al. (2004). „Benefits for Wind Energy in Electricity Markets from Using Short Term Wind Power Prediction Tools; A Simulation Study“. en. In: *Wind Engineering* 28.1, pp. 119–127. DOI: 10.1260/0309524041210838.

Vannitsem, S. (2017). „Predictability of large-scale atmospheric motions: Lyapunov exponents and error dynamics“. In: *arXiv:1703.04284 [nlin]*. DOI: 10.1063/1.4979042.

Vannitsem, S., J. B. Bremnes, J. Demaeyer, et al. (2020). „Statistical Postprocessing for Weather Forecasts – Review, Challenges and Avenues in a Big Data World“. EN. In: *Bulletin of the American Meteorological Society* -1.aop, pp. 1–44. DOI: 10.1175/BAMS-D-19-0308.1.

Vannitsem, S., D. Wilks, and J. Messner (2018). *Statistical Postprocessing of Ensemble Forecasts*. en. Elsevier. DOI: 10.1016/C2016-0-03244-8.

Veldkamp, S., K. Whan, S. Dirksen, and M. Schmeits (2020). „Statistical post-processing of wind speed forecasts using convolutional neural networks“. In: *arXiv:2007.04005 [physics, stat]*.

Wang, H. and D.-Y. Yeung (2020). „A Survey on Bayesian Deep Learning“. In: *arXiv:1604.01662 [cs, stat]*.

Weingart, N. (2018). „Deep Learning based Error Correction of Numerical Weather Prediction in Switzerland“. MA thesis. Swiss Federal Institute of Technology.

Weiss, A. (2001). „Topographic position and landforms analysis“. In: *Poster presentation, ESRI user conference, San Diego, CA*. Vol. 200.

Weusthoff, T. (2011). „Weather Type Classification at MeteoSwiss – Introduction of new automatic classifications schemes“. In: *Arbeitsberichte der MeteoSchweiz*.

Wilks, D. S. (2011). *Statistical methods in the atmospheric sciences*. Vol. 100. International Geophysics. Academic Press.

Wilson, A. G. (2020). „The Case for Bayesian Deep Learning“. In: *arXiv:2001.10995 [cs, stat]*.

Colophon

This thesis was typeset with LaTeX 2$_\varepsilon$. It uses the *Clean Thesis* style developed by Ricardo Langner. The design of the *Clean Thesis* style is inspired by user guide documents from Apple Inc.

Download the *Clean Thesis* style at `http://cleanthesis.der-ric.de/`.

Declaration

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Lamone, April 6, 2021

Francesco Zanetta