BONNER METEOROLOGISCHE ABHANDLUNGEN

Heft 96 (2023) (ISSN 0006-7156)
Herausgeber: Andreas Hense

Lukas Hubert Leufen

Time Filter Assisted Deep Learning to Predict Air Pollution

BONNER METEOROLOGISCHE ABHANDLUNGEN

Heft 96 (2023) (ISSN 0006-7156)
Herausgeber: Andreas Hense

Lukas Hubert Leufen

Time Filter Assisted Deep Learning to Predict Air Pollution

Time Filter Assisted Deep Learning to Predict Air Pollution

DISSERTATION
ZUR
ERLANGUNG DES DOKTORGRADES (DR. RER. NAT.)
DER
MATHEMATISCH-NATURWISSENSCHAFTLICHEN FAKULTÄT
DER
RHEINISCHEN FRIEDRICH-WILHELMS-UNIVERSITÄT BONN

vorgelegt von
Master of Science
Lukas Hubert Leufen
aus
Ludwigsburg

Bonn, März, 2023

Diese Arbeit ist die ungekürzte Fassung einer der Mathematisch-Naturwissenschaftlichen Fakultät der Rheinischen Friedrich-Wilhelms-Universität Bonn im Jahr 2023 vorgelegten Dissertation von Lukas Hubert Leufen aus Ludwigsburg.

This paper is the unabridged version of a dissertation thesis submitted by Lukas Hubert Leufen born in Ludwigsburg to the Faculty of Mathematical and Natural Sciences of the Rheinische Friedrich-Wilhelms-Universität Bonn in 2023.

Anschrift des Verfassers: Address of the author:

Lukas Hubert Leufen
Jülich Supercomputing Centre
Forschungszentrum Jülich
Wilhelm-Johnen-Straße
D-52428 Jülich

1. Gutachter: PD Dr. Martin Georg Schultz,
Forschungszentrum Jülich
2. Gutachter: Prof. Dr. Andreas Hense,
Rheinische Friedrich-Wilhelms-Universität Bonn

Tag der Promotion: 20. Juni 2023

Abstract

Exposure to ground-level ozone harms human health as well as the entire ecosystem, so accurate prediction of ozone exposure is of particular importance. Machine learning (ML), and deep learning (DL) in particular, has emerged as a powerful method with a vast variety of applications, including meteorology and Earth system sciences, making it a strong alternative to conventional methods such as chemical transport models (CTMs) or regression based solutions to forecast ground-level ozone. However, to date, classical as well as ML approaches have experienced challenges in reliably forecasting ozone pollution at the local scale. These shortcomings can be attributed to the challenges posed by inherent uncertainties about near-future weather conditions and the superposition of patterns on different time scales. In this thesis, a time series filtering approach to split up long-term and short-term variations and DL are applied to allow for accurate predictions of air pollution attributable to ground-level ozone. This is complemented by integrating large amounts of data from air quality monitoring stations distributed across Central Europe, climatological statistics on air pollutants and meteorological data from numerical weather models. The DL approach is framed by a well-defined workflow for training and validation called MLAir, which ensures the reproducibility of the findings. Results substantiate that the combination of sophisticated DL architectures and time series filtering enables accurate ozone prediction. The DL approach thereby achieves a nearly bias-free prediction and has a good performance with regard to the seasonal variability of ozone. This leads to a great improvement compared to simpler reference forecasts based on climatology and persistence, as well as to the Copernicus Atmosphere Monitoring Service (CAMS) regional multi-model ensemble forecast, which combines nine individual state-of-the-art CTMs deployed operationally by public weather services and research institutions. Averaged over a forecast horizon of four days, the prediction for the daily maximum 8-hour running average (dma8) of ozone by the CAMS regional ensemble has a root mean squared error (RMSE) of 7.6 ppb, whereas the newly developed method here achieves an RMSE of 5.1 ppb. The approach presented in this thesis thus marks an important advance in DL-based air pollution prediction, benefiting the general public through more reliable forecasts. Furthermore, this study opens up the prospect of further research opportunities towards the prediction of a range of other air pollutants or related applications in meteorology.

Contents

1. Introduction

1.1. Motivation

Human health is significantly shaped by the quality of the ambient air in which humans live. The World Health Organization (WHO, 2013) points out that the negative effects of air pollution on health are very well documented in many studies conducted worldwide. Short-term exposure to higher levels of air pollutants such as particulate matter or ozone increases hospital admissions (WHO, 2013) and respiratory, cardiovascular and all-cause mortality (Romieu et al., 2012; WHO, 2013). For long-term exposure, Cohen et al. (2017) report an increase in air pollution-related mortality from 1990 to 2015, with an estimated 4 million additional deaths and 103 million years of life lost attributable to particulate matter and ozone alone. Zhang et al. (2018) also report a 13% increase in ozone-related mortality between 1990 and 2010 in the United States, despite declining ozone concentrations. If ozone concentrations had remained at 1990 levels, a 55% increase in the ozone-related mortality burden would have been attributable over the same period (Zhang et al., 2018). This underlines the enormous impact of actions to improve air quality. Since air quality is determined by a complex interplay of emissions, transport as well as chemical and physical transformation processes (Solberg et al., 2016), exposure to air pollution is rarely in the hands of the individual but needs to be controlled by public authorities on the local, regional, national or international level. Good air quality, therefore, requires broad measures across a wide range of sectors, such as energy production, industry, transport, but also individual housing and lifestyles (WHO, 2013). Europe's latest air quality status report from the European Environment Agency (2022) shows that levels of ambient air pollutants are above European Union (EU) and WHO standards all over Europe. In particular, ozone pollution remains high, with 12% of the EU population exposed to ozone concentrations above EU standards and 95% above WHO guidelines, placing ozone as a ubiquitous public health hazard.

Effective countermeasures require an in-depth understanding of air quality and atmospheric chemistry through extensive monitoring and accurate modelling. This includes the need to describe the current state and provide a precise air quality prediction. Numerical and statistical models are used to understand the spatial and temporal distribution and evolution of ozone concentrations and chemistry. The most comprehensive and widely used models are so-called chemical transport models (CTMs), which are based on a complex combination of physical and chemical processes to simulate the distribution and evolution of chemical compounds from global to local scales. Statistical methods, however, are applied more in applications at the local scale, where they mainly map correlations between observed weather and air quality, for example, through linear regression, without explicitly solving physical and chemical equations, but rather are based primarily on data. Furthermore, statistical methods are widely applied to correct biases, e.g. for CTM output. While CTMs are set up very broadly to describe atmospheric chemistry comprehensively, statistical methods are always very targeted to a specific objective.

Seltzer et al. (2020) remark that estimates of air pollution exposure and effects may differ significantly between studies based on different CTMs. According to Vautard et al. (2012), uncertainties in CTMs derive first from chemical and aerosol physics, second from fluxes such as emissions and deposition, and third from meteorological processes that influence transport, but also surface fluxes or chemistry. Furthermore, the level of detail of the simulated processes always depends on the intended use case of the model and is constrained by the available computing capacity (Young et al., 2018). Uncertainties from meteorological processes cause inconsistencies in the sensitivity of the air quality modelling to important variables such as temperature, relative humidity, solar irradiance, precipitation and the height of the planetary boundary layer (Brunner et al., 2015; Otero et al., 2018). Thus CTMs tend to overestimate, for example, the influence of maximum temperature and radiation, both of which are directly coupled to ozone production, and, on the other hand, poorly represent the influence of moisture, which is crucial in ozone reduction processes (Otero et al., 2018). In addition, under stable conditions, wind speed is often exaggerated, which leads to an underestimation of primary pollutants due to exaggerated air mixing (Vautard et al., 2012; Bessagnet et al., 2016).

In many CTMs, meteorological fields calculated in advance are used to simulate the chemistry so that the chemical processes taking place can be studied in more detail and compared with observations (Young et al., 2018). However, there are also modelling approaches in which the meteorological and chemical processes are simultaneously simulated. In both approaches, a large share of the computational effort is spent on calculating the chemical kinetics alone since coupled ordinary differential equations have to be solved repeatedly. This high effort is a result of the chemical reaction equations that are non-linear, strongly coupled and stiff and therefore require an expensive numerical integration scheme. Thus, the more substances and reactions are considered, the more computational effort is inevitable (Wang et al., 1999). To reduce the computational effort, the chemical mechanisms are necessarily simplified, especially in global models. However, this also simplifies the tropospheric gas phase oxidation, especially for organic molecules and their oxidation pathways, leading to uncertainties in the overall modelling of tropospheric chemistry (Young et al., 2018).

Additionally, global models are typically run at a coarser resolution due to computational resource constraints to allow for longer-range modelling, for example. However, Stock et al. (2014) note that this can lead to inaccuracies for ground-level air pollutant concentrations on the local scale since atmospheric chemistry and chemical substances become relevant on scales smaller than the model resolution, whereas they cannot be simulated sufficiently. Young et al. (2018) state that an increase in spatial resolution offers the potential for CTM results to become more representative of local measurements. However, according to their assessment, this is not yet computationally feasible on the global scale. In fact, improving resolution provides a better representation of local chemistry but increases computational costs without necessarily improving the simulation of global-scale chemistry (Stock et al., 2014). Schaap et al. (2015) argue that an improvement of the spatial resolution leads to a better simulation of the spatial distribution during air pollution episodes, but at the same time, temporal variability is not improved. Accordingly, simulations in urban areas can become more accurate, but an adjusted spatial resolution does not lead to improvements for rural regions, which are more influenced by large-scale processes. In fact, according to Schaap et al. (2015), increasing resolution can degrade the representation of spatial contrasts between metropolitan areas. Colette et al. (2014) name the availability and resolution of auxiliary parameters for calculating key emissions as a further limitation, especially when a CTM is used in operational forecasting. Schaap et al. (2015), therefore, propose to determine an optimal resolution that balances the improved simulation on the one hand and the additional computational effort on the other. Finally, a higher resolution also depends on the technical implementation and the degree of concurrency and thus the scalability of the model code, since adding computing power, for example, by additional central processing units (CPUs) also increases the amount of inter-node communication (Baklanov et al., 2014).

To link models and observations and to deal with the uncertainties and limitations of CTMs, one possibility is to use post-processing methods as applied in Fuentes and Raftery (2005). The authors use a posterior distribution over the model bias to estimate the actual value in dependence of the model prediction and the measured value at the observing air quality station (AQS). An alternative approach is to combine different CTMs in an ensemble, as favoured by Manders et al. (2012), especially since various studies show that there are fewer differences between individual simulations of the same model than between simulations with different models and model setups (Brunner et al., 2015; Bessagnet et al., 2016; Seltzer et al., 2020). A further common practice to address this issue is the use of statistical methods such as lasso, logistic or multiple linear regressions, as used, for example, in Otero et al. (2016) and Jahn and Hertig (2021). These techniques obtain valuable information for air quality prediction from meteorological parameters such as mean air temperature and geopotential heights at 850hPa, but also from the persistence of air pollutants as well as maximum ground-level temperature and humidity.

However, despite decades of research, it is still not possible to make reliable, bias-free predictions of ground-level pollutants. The roots of these problems are the limitations in modelling but also the pronounced day-to-day and intra-day variability of pollutant concentrations resulting from the complex relationship between meteorology and atmospheric chemistry (Manders et al., 2012). The daily to weekly variations in air pollution are also amplified by covariation on the synoptic time scale (Fiore et al., 2015). For instance, tropospheric ozone has a lifetime of one to two months in the free troposphere and a few weeks near the surface, with shorter lifetimes during summer. Short lifetimes result in high spatial and temporal variability (Junge, 1974). The presence of patterns on different temporal scales makes the analysis and interpretation of the time series more challenging, as the annual cycle of ozone is not regular but varies from year to year, while local meteorological conditions strongly affect the day-to-day variability (Eskridge et al., 1997; Wise and Comrie, 2005). This results in unique events with peak concentrations that are very difficult to predict due to their complex origin (Wise and Comrie, 2005).

Recently, the use of emerging artificial intelligence (AI) approaches such as machine learning (ML) and deep learning (DL) have seen increasing attention in meteorology and Earth system science (c.f. Reichstein et al., 2019; Dueben et al., 2022). ML approaches offer the potential to improve forecasting as they can understand and learn non-linear relationships. ML refers to methods that enable a machine to detect and reproduce a context in a training process by feeding in input data without being explicitly programmed. The fields of ML application can support or replace classical parametric and statistical approaches, ranging from classification tasks and anomaly detection, such as extreme weather patterns or land use and its change, to regression problems, such as prediction of fluxes or vegetation properties based on atmospheric conditions, and to state prediction, such as the short-term precipitation forecast, downscaling and bias correction of forecasts or seasonal forecasts (see Reichstein et al., 2019). Schultz et al. (2021) also discuss whether DL can replace numerical weather prediction in the near future and conclude that end-to-end DL weather forecast applications, in particular, harbour great potential, as they can be tailored to a specific problem.

DL methods are a subgroup of ML methods that use many levels of representation consisting of a composite of several simple but non-linear modules. These modules compute a simple input-output mapping. By linking a sufficient number of such modules in layers, it is possible to obtain different degrees of abstraction levels, so that very complex functions and relationships can be learned (LeCun et al., 2015). The term *deep* refers to the fact that DL methods make use of a large number of representation layers. However, there is no clear definition of how many layers are considered deep, resulting in a lot of misused terminology in the literature to sell a study as deep learning. Moreover, boundaries also seem to be shifting more and more as both the number of layers and the total number of parameters continue to increase. For example, in 1989 the largest models consisted of four layers and had a total of a few thousand parameters (LeCun et al., 1989). In 2015, by contrast, modern architectures reached 10 to 20 layers with hundreds of millions of parameters (LeCun et al., 2015). Nowadays, the deepest models as the GPT-3 model consist of more than 175 billion parameters distributed over up to 96 layers (Brown et al., 2020). DL in particular has led to impressive advances in computer vision and speech recognition.

The problems from these fields and the DL methods to solve them may be transferred to meteorological applications (Schultz et al., 2021). However, for example, standard computer vision applications deal with images of three colour channels. In contrast, meteorological problems are more oriented towards multivariate problems, where the correlation and causality between different channels or variables may be different and change over time. Similarly, although the analogy between frequency patterns in speech recognition and the variation of atmospheric time series is evident at first glance, atmospheric time series are permeated by autocorrelation and important features overlapping at very different time scales, which poses fundamental difficulties for DL methods according to Cui et al. (2016). Therefore, Reichstein et al. (2019) argue that conventional ML methods may not be ideally suited to specific problems in Earth system science because, among other things, spatial and temporal context at different scales needs to be better accounted for in the methods. Schultz et al. (2021) accordingly conclude that novel DL methods are needed due to the special properties of weather data. Given these fundamental problems of applying DL in meteorology and the specific challenges posed by the complexity of the processes driving air quality, DL approaches have so far experienced difficulties in reliably reproducing and accurately predicting air pollution at the local scale. In the past year of 2022, impressive results were indeed achieved in DL-based weather forecasting. In chronological order, the models FourCastNet (Pathak et al., 2022), Pangu-Weather (Bi et al., 2022) and GraphCast (Lam et al., 2022) stand out in particular, as they perform on par with or even better than operational weather forecasting models. However, all these approaches focus on global weather forecasting, and there have been no attempts to use these methods for air quality forecasting.

More than 20 years ago, Cobourn et al. (2000) described the difficulties that local air pollution control agencies have in finding a comprehensive methodology to predict ozone, as it is not possible to quantify the relative effectiveness of research results for a number of reasons. The multi-faceted reasons are, inter alia, differences from study to study in terms of time and location covered, different choice of predictive parameters, statistically unrepresentative data, or the use of inaccessible parameters at the time a forecast is issued such as morning NO levels, which themselves require a prediction and are therefore prone to uncertainties. Furthermore, as Cobourn et al. (2000) argue, many prediction models perform inadequately for ozone prediction as they are too basic in design. As the following sections show, the issues of reproducibility and applicability continue to be a central concern in the scientific discourse. Even though methods for ozone prediction have evolved, models for reliable air quality prediction are still lacking. In fact, it is a fallacy that it is sufficient to use more and more modern and complex models like DL for the same research questions and to expect that this will automatically lead to an improvement.

The purpose of this research is therefore to explore several questions related to the prediction of ground-level ozone on local scale based on DL methods to provide a reliable ozone forecast for a couple of days into the future. In particular, this thesis addresses the shortcomings of previous DL approaches that arise from the variability and superposition of different scales of ozone. Special attention is also dedicated to the reproducibility of the results since the foundation of a reliable ozone forecast is that the research work that led to the results is presented transparently and reproducible. The remainder of this chapter provides an overview of the theoretical background of ground-level ozone in its role as an air pollutant and how ozone is modelled conventionally (Section 1.2). Then I move on to a survey of DL methods and their application for time series and ozone prediction (Section 1.3). As a subtopic, I address issues regarding reproducibility in science and DL research and discuss how to ensure that results obtained using DL are verifiable and reproducible for the scientific community (Section 1.3.4). Finally, in Section 1.4, I specify the research questions of my thesis and give an outlook on the following chapters.

1.2. Ground-Level Ozone as Air Pollutant

Ozone near the ground is fundamentally classified as an air pollutant because exposure to ozone leads to respiratory and cardiovascular effects for humans and also causes plant damage (WHO, 2013; US EPA, 2020). Ozone impacts the human body in a vast number of complex ways. When inhaled, ozone reacts with lipids, proteins and antioxidants in the respiratory tract, resulting in the formation of secondary oxidation products, which in turn cause a number of physiological reactions. Initial indications are, for example, inflammation of the lungs. Such first physiological reactions can trigger a variety of autonomic, endocrine, immune and inflammatory system-wide reactions at the cellular, tissue and organ levels (US EPA, 2020). Thereby, short-term ozone exposure has an age-dependent negative impact, so older groups of people are particularly exposed (Romieu et al., 2012; Bell et al., 2014). In addition, Bell et al. (2014) report a higher risk potential for persons unemployed or with lower occupational status, and persons with lower education or those living in poverty also seem more at risk of being affected by ozone.

Besides the effects on humankind, ozone also impacts plants and entire ecosystems. According to Mills et al. (2018), there is evidence that the higher the ozone concentration, the greater the likelihood of crop yield decline and growth inhibition. For example, ozone has been found to cause loss of stomatal control, incomplete nighttime stomatal closure, and decoupling of photosynthesis and stomatal conductance. These impacts, in turn, have negative consequences at community and ecosystem scales, which can be noticed in the species composition (US EPA, 2020). Though Mills et al. (2018) show that ozone metrics relevant to crop impact can be high in humid and dry, cooler or warmer regions, ozone-related damage to wheat, for example, is particularly pronounced in tropical regions (Shindell et al., 2019). Overall, the highest exposure of plants to ozone occurs in regions of the world where high emissions and climatic conditions combine to foster the formation of ozone. Examples include the southern regions of the US, southern Europe, northern India and northwestern and eastern parts of China (Mills et al., 2018).

1.2.1. Chemistry of Tropospheric Ozone

The particularity of ozone (O_3) compared to other air pollutants, such as particulate matter or nitrogen oxide (NOx), is that O_3 is a secondary air pollutant. This means that O_3 itself is not emitted to the atmosphere directly but is a result of chemical reactions with precursors in the atmosphere (Monks et al., 2015). Accordingly, the O_3 concentration in the surface boundary layer is mainly driven by three key processes; (1) photochemical reactions leading to production and destruction, (2) atmospheric transport of O_3 and its precursors, and (3) loss of O_3 and interacting chemical substances from dry and wet deposition (Young et al., 2018). Basically, the production of O_3 is mainly related to the photolysis of nitrogen dioxide (NO_2) in the presence of sunlight with wavelengths below 424 nm (Seinfeld and Pandis, 2016, p.179)

$$\mathrm{NO_2} + h\nu \longrightarrow \mathrm{NO} + \mathrm{O}\,. \tag{R1}$$

The product O of this process associates with the oxygen molecule (O_2) in a termolecular reaction with a third co-reactant M to form O_3 (Monks et al., 2015)

$$\mathrm{O} + \mathrm{O_2} + \mathrm{M} \longrightarrow \mathrm{O_3} + \mathrm{M}\,. \tag{R2}$$

In the presence of sufficient NO, the O_3 thus formed reacts with the NO and produces NO_2 (Seinfeld and Pandis, 2016, p.179)

$$\mathrm{O_3} + \mathrm{NO} \longrightarrow \mathrm{NO_2} + \mathrm{O_2}\,. \tag{R3}$$

The conversion between NO and NO_2 is a rapid process, with a photolysis rate of about $10^{-2}s^{-1}$, so both substances are grouped as NOx (Monks et al., 2015). Moreover, since the concentration of O_2 can be regarded as constant, the O_3 concentration is determined solely by the ratio of the concentration of NO_2 to NO (Seinfeld and Pandis, 2016, p.180)

$$[\mathrm{O_3}] \sim \frac{[\mathrm{NO_2}]}{[\mathrm{NO}]}\,. \tag{1.1}$$

However, (R1)-(R3) alone cannot explain measurements of O_3 concentrations. For example, in heavily polluted regions, NO, in particular, is emitted, which leads to the destruction of O_3 after (R3), but at the same time, a high O_3 load might be measured (Seinfeld and Pandis, 2016, p.180). According to Levy (1971), the primary factor is the hydroxyl radical oxidation of carbon monoxide (CO), methane (CH_4), and non-methane volatile organic compounds (NMVOCs). The reaction of CO with OH results in CO_2 and a hydroperoxyl radical (HO_2)

$$CO + OH \xrightarrow[O_2]{} CO_2 + HO_2\,. \tag{R4}$$

The hydroperoxyl radical is more reactive than pure O_2 and reacts in presence of NO, leading to the formation of NO_2 and OH

$$HO_2 + NO \longrightarrow NO_2 + OH\,. \tag{R5}$$

(R4) followed by (R5) leads to a shift in the NOx ratio so that the steady-state concentration of O_3 increases according to (1.1). The decisive factor here is the NOx concentration, which directly influences (R5). At low NOx concentrations, O_3 production increases linearly with NO concentration and proportionally to the HOx production rate. However, in a high NOx regime, O_3 production increases linearly with the CO concentration as well as the HOx production rate, but at the same time also decreases with increasing NOx concentration, which can eventually lead to a decrease in O_3 (Seinfeld and Pandis, 2016, pp. 182-184). In the case of the oxidation of CH_4, CO is also formed as the primary product, which in turn leads to (R4). Apart from their complexity due to the size of the molecules and thus strongly simplified, the reaction chains of NMVOCs are similar to the processes presented here regarding the initiation by OH and the formation of O_3 by the reaction of peroxy radicals and NO (Seinfeld and Pandis, 2016, pp. 188-192). So, CH_4 and NMVOCs are grouped as volatile organic compounds (VOCs).

Since the chemical conversions, as well as transport and deposition, depend on the atmospheric conditions, O_3 concentrations show a pronounced variability both in the course of the day and from day to day (Manders et al., 2012). For photochemical processes as well as for the emission of VOCs, radiation and temperature play a central role. Likewise, dry deposition is influenced by radiation and temperature but also wind speed and humidity. Wet deposition of chemical compounds affecting the ozone chemistry is mainly driven by precipitation intensity and type (Vautard et al., 2012). Furthermore, the regional and local weather controls the transport between regions and within a region, so the variation of O_3 on scales of days to weeks is determined by the large-scale high and low-pressure systems (Fiore et al., 2015). Finally, O_3 concentration due to the chemical conversions is strongly dependent on anthropogenic emissions of NOx, e.g. from power generation and transportation, and natural sources such as wildfire, lightning or soil, with the former dominating in urban regions (Russell et al., 2012). The emission of VOCs from anthropogenic or natural sources also significantly determines the O_3 level (Porter et al., 2017). For example, the analysis by Guo et al. (2018) reveals that the highest O_3 concentrations in the U.S. were largely caused by emissions of VOCs. And lastly, the exchange of stratospheric O_3

into the troposphere has an effect on the O_3 concentration, whereby this contribution is lower than the tropospheric production by a factor of 5-7 (Zhang et al., 2016).

1.2.2. Temporal Variability of Ozone

Data in meteorology and atmospheric chemistry usually consist of a sequence of observations or model results. If the ordering is an essential property, data are referred to as time series. If the statistical properties of a time series remain constant over time, a time series is called stationary. A distinction is made between strict and weak stationarity. Strict stationarity applies when the joint distribution does not change over time. Weak stationarity, on the other hand, only requires that the mean value is constant over each sample period and that the covariance between different samples of the time series only depends on their relative position to each other, but not on their absolute position. Many statistical methods assume weak stationarity of a time series as a basic requirement (Wilks, 2006). However, atmospheric time series consist of a superposition of patterns on different time scales and exhibit pronounced cycles such as the seasonal and diurnal cycle, so stationarity cannot be assumed. Moreover, the seasonal cycle is not regular but varies substantially from year to year (Eskridge et al., 1997).

Wilks (2006) mentions two methods to deal with non-stationarity so that the result can be considered stationary. One is to stratify the data, i.e. divide them into smaller homogeneous subsets sharing similar statistical properties, for example, by season or even smaller subsets into monthly blocks, and carry out analyses on each subset separately. However, the results of stratified sampling are not inherent to other samples of a time series in general. The other approach is to use periodic averaging methods to remove the seasonal variation from the data so that the resulting time series has a mean of zero. Rao and Zurbenko (1994), for example, understand a time series $X(t)$ as the sum of a trend component $e(t)$, a seasonal variation $S(t)$, and stochastic component $W(t)$

$$X(t) = W(t) + S(t) + e(t) \ . \tag{1.2}$$

There are several works looking for a suitable separation method. Rao and Zurbenko (1994) use a so-called Kolmogorov-Zurbenko filter (KZF), which is a low-pass filter realised by iterating a moving average multiple times on the time series. Yang and Zurbenko (2010) note that Zurbenko (1986) previously compared different types of finite impulse response (FIR) filters, which are realised by a convolution of the time series with a window function (Oppenheim and Schafer, 1975) such as the Bartlett window or Tukey-Hamming window, where the KZF window was closest to the optimal decomposition. Eskridge et al. (1997) show that a simple calculation of the anomaly of a time series cannot adequately separate the synoptic and seasonal signals. The study by Hogrefe et al. (2003) finds equal performance of KZFs, wavelet and Fourier transforms, and elliptic filters. However, they note that elliptic filters introduce a phase shift in the signal, making it difficult to interpret the signals in a meaningful way. Rao et al. (2020) additionally compare a more advanced empirical mode decomposition method but find no improved decomposition properties. Lastly, Meyer et al. (2021) use a more simplistic sinusoidal curve to extract seasonality. However, the parameters of this curve can only be fitted retrospectively, so year-to-year variability cannot be accounted for. It can therefore be summarised that there are a large number of different methods for decomposing time series and that there is no method that is proven to be superior to the others. Also, the appropriate choice of method may depend on the application and cannot be answered generally.

In meteorology and atmospheric chemistry, different physical phenomena cause a range of processes with various frequencies, so a decomposition according to the dominant temporal scales is not only necessary from a statistical point of view but also scientifically meaningful (Rao et al., 1997). Hence, Rao et al. (2011) interpret the concentration of ozone as a modulation of a baseline, where the baseline is driven by climate and long-term emissions, and the modulation is driven by weather. Rao et al. (1997), as well as Eskridge et al. (1997), separate the temporal scales of e, S and W as in (1.2) based on two cut-off periods of 33 days and 1.7 years

$$X(t) = W(t)\Big|^{33d} + S(t)\Big|^{1.7y}_{33d} + e(t)\Big|_{1.7y} . \tag{1.3}$$

Vertical lines indicate the upper and lower cut-off period for a particular component; empty entries denote an open limit. Meanwhile, Hogrefe et al. (2003) identify five time scales in particular relevant to ozone

$$X(t) = ID(t)\Big|^{11h} + DU(t)\Big|^{2.5d}_{11h} + SY(t)\Big|^{21d}_{2.5d} + SE(t)\Big|^{2.5y}_{21d} + LT(t)\Big|_{2.5y} . \tag{1.4}$$

The intraday (ID) component includes the most rapidly proceeding events and local-scale processes with periods below 11 hours. The diurnal (DU) component with periods up to 2.5 days is related, in particular, to the differences between day and night of the meteorological and chemical processes. The synoptic (SY) component (periods up to 21 days) is determined by the changing weather events on the trans-regional scale. Variations with a period up to 2.5 years can be attributed to the seasonal (SE) component and are related to the seasonal changes. Finally, all effects with periods longer than 2.5 years are grouped as long-term (LT) variations, which include, for example, interannual variability and trends due to climate or policies.

Other studies, such as those by Galmarini et al. (2013) or Kang et al. (2013), follow the basic approach of Hogrefe et al. (2003), but combine all patterns with a period greater than 21 days as a baseline LT component rather than splitting them, so that (1.4) becomes

$$X(t) = ID(t)\Big|^{11h} + DU(t)\Big|_{11h}^{2.5d} + SY(t)\Big|_{2.5d}^{21d} + LT(t)\Big|_{21d} . \tag{1.5}$$

According to Galmarini et al. (2013), more than half of the variance of ozone can be attributed to the DU component, making it the central driver of ozone variability. In addition, LT and SY are of secondary relevance in explaining the variance, with the importance of SY changing, in particular, seasonally and with wind direction. The ID component has only a minor fluctuation and is considered the least influential in the overall variation of ozone, acting more as noise.

It is very challenging to find a suitable technique that can cleanly decompose a time series (Kang et al., 2013). A good separation technique concentrates the energy of a relevant time scale in one component each and does not distribute it among multiple components (Rao et al., 1997). However, the individual components of a signal decomposition of environmental time series are not orthogonal to each other, so a clear separation cannot be achieved, and the individual components remain correlated (Kang et al., 2013). For example, Galmarini et al. (2013) show that only 80% of the explained variance can be attributed to the individual shares of the four components, and the remainder arises from the interaction between the different scales. Considerably clearer separation is possible, though, according to Kang et al. (2013), when ID, DU, and SY from (1.5) are combined into a single short-term (ST) component

$$X(t) = ST(t)\Big|^{21d} + LT(t)\Big|_{21d} . \tag{1.6}$$

Again, this agrees with the basic idea of Rao et al. (1997) that ozone time series can be understood as a modulation represented by ST of a baseline given by LT. The disadvantage of this approach is that the ST component covers a wide range of scales and no longer allows discrimination between local and synoptic processes.

1.2.3. Modelling Ozone at Local Scales

For accurate simulation of, for example, ozone exposure on local scale, it is important that models capture the baseline concentrations (Kang et al., 2013) but also the seasonal and diurnal variations of ozone in particular (Seltzer et al., 2017). The study of Solazzo et al. (2017), who apply a decomposition of the error according to the motion of scales, reveals that especially the long-term bias and the diurnal fluctuations contribute to the error of CTMs. Otero et al. (2018) show systematic deviations in reflecting the seasonal and diurnal cycles between different CTMs when compared to observations. According to their findings, CTMs generally overestimate, for example, ozone globally. Guo et al. (2018) show that CTMs have a positive bias in simulating ozone and furthermore the timing of events of highest ozone levels is not well represented. In contrast, Im et al. (2015) report on a tendency of CTMs to overestimate lower ozone concentrations, whereas high values are sometimes severely underestimated, leading to an underestimation of ground-level ozone across the year. Likewise, Manders et al. (2012) find that CTMs are not able to reproduce peak concentrations. Even though Bessagnet et al. (2016) also show an underestimation of ozone concentrations for some periods, they conclude that CTMs mostly overestimate the observed ozone concentrations, which is in agreement with the research by Vautard et al. (2012) and Young et al. (2018).

Therefore, a variety of alternative approaches are used for the prediction of ozone. Cheng et al. (2022) provide an in-depth overview and highlight the advantages and disadvantages of air quality prediction using physically based methods such as CTMs, simple empirical approaches such as persistence or climatology, and parametric and statistical methods such as decision trees, regression approaches or neural networks. However, the use cases are very heterogeneous, so only some methods and applications can be directly contrasted. The prediction setups may differ, as shown subsequently. Therefore, let t_0 be the current time step, while any time step t_i relative to t_0 falls in the future if $t_i > t_0$, and in the (relative) past in case of $t_i < t_0$.

- Observed or modelled weather data for past time steps ($t_i \leq t_0$) are used to predict ozone at subsequent time steps ($t_i > t_0$).
- The forecast relies on weather forecasts in the forecast horizon $t_i > t_0$ to derive ozone at the corresponding time steps ($t_i > t_0$).
- Ozone for a current time t_0 is predicted based on the current weather data (t_0) while neglecting the temporal development and context.
- Purely time series-based approaches predict future ozone ($t_i > t_0$) solely based on the history of ozone ($t_i \leq t_0$) or other air pollutants.

On top of the high level of variation in the application, there is also no common agreement in the scientific community on which meteorological or chemical variables should be used as predictors of ozone. An analysis of the most important predictors can be found in Otero et al. (2016) and Jahn and Hertig (2021), for example. Most consensus is reached on the importance of meteorological variables such as temperature, humidity, pressure, cloud cover and wind. However, some studies use data near the surface, such as measurements or the lowest level in a weather model, while other work is based on data from elevated layers in weather models, e.g. at 850 hPa or 500 hPa. Some studies include radiation, geopotential height, boundary layer height, and metadata such as population density, soil type, altitude, and the hour of the day or day of the year. Finally, some research studies use chemical quantities as additional input values. Thereby, lagged ozone is a powerful predictor (see Jahn and Hertig, 2021), especially under stationary conditions with consistently high or low air pollution. However, under such conditions, persistence alone is actually a very good predictor, as Zhang et al. (2012) note. Accordingly, a method overly dependent on lagged ozone can have reduced predictive performance, especially at the beginning and end of such periods. Furthermore, particular care must be taken when using chemical species, as chemical substances have a high degree of interconnectivity. Cheng et al. (2022) point out that, for example, NO_2 is not suitable as a predictor in the forecast horizon $t_i > t_0$ because NO_2 and ozone are strongly coupled. It should be noted that there are several such studies, and for the reason given, they are not considered further.

Applications of simple linear regression can be found, for example, in Cheng et al. (2007) and Demuzere and van Lipzig (2010). To support the simple methods, Cheng et al. (2007) classify the large-scale weather situation in advance using principal component analysis. Demuzere and van Lipzig (2010) choose a similar approach based on an automated Lamb weather classification. In Jahn and Hertig (2021), lasso regression is used in addition to multiple linear and logistic regression. Otero et al. (2016) also use multiple linear regressions and additionally call on logistic regression to quantify the probability of threshold exceedance. Similarly, Cobourn and Hubbard (1999) use a hybrid model consisting of a standard non-linear regression plus a model for particularly high and low values. Munir et al. (2012) favour a quantile regression model to better represent the deviation of the ozone distribution from normal. Other common classical approaches are generalised linear models, such as in Camalier et al. (2007) and Wells et al. (2021), and generalised additive models, such as those found in Schlink et al. (2006), Pearce et al. (2011) and Gao et al. (2022). Examples of simple methods from the field of ML are tree-based approaches such as random forest in Siwek and Osowski (2016) and Zhan et al. (2018) or classification and regression trees as in Ryan (1995). Besides random forest models, Gao et al. (2022) also use support vector regression methods.

This section provides a brief overview of classical approaches to local air quality prediction. A disadvantage of most methods is that they explicitly assume a normal distribution of the target variable or cannot deal with the non-linear relationships in ozone chemistry, contrary to neural networks. Accordingly, early work by Comrie (1997) and Cobourn et al. (2000) shows that using neural networks can produce equivalent or even improved ozone predictions. With the growing amount of available data and the increasing impact of advanced ML approaches, especially DL, there is great potential for local ozone modelling. An overview of DL applications for ozone prediction is given in Section 1.3.3, as some DL concepts require introduction first.

1.3. Deep Learning for Atmospheric Science

Classical approaches often follow the principle of calculating an output with the help of a computer from data and an algorithm derived from domain knowledge, for example. Conversely, ML follows the principle of presenting the computer with data and a target value so that it can independently discover and reproduce the relationship. Therefore, an elementary element in ML is the training process. With regard to DL, the training follows straightforward ideas. By assembling simple parameterised modules consisting of linear or pointwise nonlinear operations, complex functions can be efficiently expressed in a multi-layer computational graph so-called neural network (NN). A relationship can then be learned based on examples from data by tuning the parameters, minimising an objective function $\mathbf{L}$, also called loss function, with a gradient-based method. This gradient can be automatically and efficiently computed using a backpropagation algorithm. Thereby, the backpropagation algorithm is simply the application of the chain rule to calculate the partial derivatives of the objective function depending on all parameters used in the DL model by backward propagating a signal across the NN (LeCun, 2019). The objective function, considered over all training examples, represents a hilly landscape in the high-dimensional optimisation space of the NN's parameters, with the direction of steepest descent given by the negative gradient that leads to a state where the error expressed by the objective function is smallest. Since hundreds of millions of parameters are adjusted in a typical NN, there is always the risk that an NN does not learn the correlations but rather learns the data itself by heart, referred to as overfitting. In order to verify the ability of generalisation, it is therefore essential to evaluate the system adapted to the data after the training on new data that were formerly unknown to the model, referred to as test data (LeCun et al., 2015).

This description applies to so-called supervised learning, which is the focus of this thesis. However, it should be mentioned that there are other DL methods where training is not supervised by a target variable. However, so-called unsupervised or semi-supervised learning methods will not be discussed further here. Since 2013, four key developments have contributed to the enormous popularity of DL methods in industry and academia. First, there are improved and novel methods; second, more and larger data sets are becoming available, allowing larger models to be trained; third, the advent of graphical processing unit (GPU) computing has massively accelerated the training of DL methods; and fourth, mature open-source software libraries with interpreted language frontends are available, making DL more accessible and deployable (Raina et al., 2009; LeCun, 2019).

To train an NN, there are several learning algorithms which address different issues and shortcomings. Moreover, in practice, it actually turns out that the best optimisation algorithm is not necessarily the best learning algorithm (Bottou and Bousquet, 2007). An appropriate choice of learning algorithm and its parameters must be reached by trial and error. This process is therefore called tuning the hyperparameters, the optimisation of the parameters of the learning algorithm over several training cycles. Nowadays, hundreds of different methods are available, each with its own hyperparameters. The choice of a suitable optimiser is therefore seen as the most delicate and, at the same time, challenging design decision (Schmidt et al., 2021). Schmidt et al. (2021) also note that for many optimisation algorithms, only empirical evaluation from the original research paper is available. Rather, since the performance of individual methods can differ substantially between tasks, it seems to the authors that the appropriate choice is more of an incessantly fluctuating state-of-the-art, which is primarily driven by hype. So LeCun et al. (2015) observe that most ML researchers fall back on the so-called stochastic gradient descent (SGD) method at that time. Thereby, a specific number of training samples, called a batch, is processed through the NN, the resulting error is calculated and propagated backwards, and the weights are adjusted accordingly. Stochastic comes into place here because each batch represents a noisy representation of the average gradient over the entire data (LeCun et al., 2015).

Before SGD gained currency, it was assumed for a long time that gradient descent methods were impractical for use in training NN, as these methods would always get stuck in local minima with poor performance (LeCun et al., 2015). However, practical and theoretical investigations by Dauphin et al. (2014) and Choromanska et al. (2015) show that it is not the poor local minima but the proliferating number of multidimensional saddle points that causes problems. Contrary to popular understanding, primarily based on lower-dimensional intuition, local minima are rare in multidimensional space (Dauphin et al., 2014). According to Dauphin et al. (2014), these saddle points are wrongly considered to be local minima because the saddle points are surrounded by high-error plateaus, which leads to a severe slowdown of the training. However, for very small NNs, there is a higher probability of finding a poor local minimum, but this chance is rapidly decreasing with the size of the NN (Choromanska et al., 2015). Choromanska et al. (2015) also prove that it becomes increasingly difficult to discover the global minimum at all for large-scale NNs, but that this does not matter because, on the one hand, the saddle points are located in a well-defined band with no difference in performance in terms of loss, and, on the other hand, this region of saddle points is preferable as finding the global minimum in practical applications is more likely to lead to overfitting the training data. After all, it is not the optimisation performance but the generalisation performance that is relevant (Bottou and Bousquet, 2007).

Besides SGD, adaptive methods like the Adam algorithm by Kingma and Ba (2014) also find a broad user base for training an NN. Adaptive methods use past gradient information to adjust the learning rate dynamically in order to achieve rapid convergence during training. Adam works well with sparse gradients, is well suited for noisy problems, is computationally efficient and requires little memory, making Adam ideal for large learning tasks in terms of the number of data or parameters (Kingma and Ba, 2014). However, Wilson et al. (2017) observe that adaptive methods achieve worse generalisation than SGD. Although adaptive methods initially show faster training advances, performance on the test data quickly stagnates. Wilson et al. (2017), therefore, recommend strongly reconsidering the application of adaptive methods. For Sivaprasad et al. (2020), meanwhile, the choice of the optimiser is double-layered, considering how well the optimiser performs absolutely, but also how difficult it is to find the optimal hyperparameter configuration. Even if SGD can yield the best performance in several cases, the configuration is very laborious to find. On the other hand, Adam often performs similarly well, and it is easier to find such a configuration. This is in line with Schmidt et al. (2021), who observe that the Adam algorithm performs decently despite many alternatives. Specifically, the authors show that optimisers consistently perform better or worse for specific architectures or tasks and that the choice of the optimiser is, therefore, dependent on exogenous factors. From a practical perspective, it is particularly interesting to note that Schmidt et al. (2021) found that hyperparameter tuning of a chosen optimiser is as effective as searching for the optimal optimisation algorithm. Thus, there is no free lunch when choosing a learning algorithm, and selecting the most suitable optimiser always involves a considerable amount of effort.

1.3.1. A Survey of Deep Learning Methods

Feedforward Neural Networks The most elementary NNs are so-called feedforward neural networks (FNNs), also known as fully-connected networks. Since there are countless different notation schemes and descriptions for NNs, the introduction and notation here largely follow that of Borovykh et al. (2018). In an FNN, all nodes of a layer are connected to all nodes in the follow-up layer. FNNs consist of $L \in \mathbb{N}$ layers with $M_l \in \mathbb{N}$ hidden nodes in each layer $l = 1, \dots, L$. Given an input $\mathbf{X} = x(0), \dots, x(J-1)$ with $\mathbf{X} \in \mathbb{R}^J$ and $J \in \mathbb{N}$, the FNN computes linear combinations of the inputs in the first layer with

$$a^1(i) = \sum_{j=0}^{J-1} w^1(i,j) \cdot x(j) + b^1(i), \quad \text{for } i = 0, \dots, M_1 - 1\ , \tag{1.7}$$

where $w^1 \in \mathbb{R}^{M_1 \times J}$ are the so-called weights and $b \in \mathbb{R}^{M_1}$ the biases. The linear combinations $a^1(i) \in \mathbb{R}^{M_1}$ are then processed by a non-linear activation function $\sigma(\cdot)$ to finally yield the outputs $f^1 \in \mathbb{R}^{M_1}$, also called activation, of the respective nodes in the first layer given by

$$f^1(i) = \sigma\left(a^1(i)\right), \quad \text{for } i = 0, \dots, M_1 - 1\ . \tag{1.8}$$

This function σ gives the FNN the ability to learn non-linear relationships in the data. Commonly used activation functions are the hyperbolic tangent (tanh) given by

$$\tanh x = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{1.9}$$

with e being the exponential function base on Euler's number, and the rectified linear unit (ReLU) function

$$\mathrm{ReLU}(x) = \max(0,x) \ , \tag{1.10}$$

but there are many more functions used as activation functions. In the following layers $l = 2,\dots,L-1$, the outputs of the preceding layer $f^{l-1} \in \mathbb{R}^{M_{l-1}}$ are always used as input from which the linear combinations are calculated on and passed through the non-linear function

$$f^l(i) = \sigma\left(\sum_{j=0}^{M_{l-1}-1} w^l(i,j)\, f^{l-1}(j) + b^l(i)\right), \quad \text{for } i = 0,\dots,M_l - 1 \ , \tag{1.11}$$

where $f^l \in \mathbb{R}^{M_l}$, $w^l \in \mathbb{R}^{M_l \times M_{l-1}}$ and $b^l \in \mathbb{R}^{M_l}$. The outputs in the last layer $l = L$ represent the outputs $\hat{y} \in \mathbb{R}^{M_L}$ of the NN

$$\hat{y}(i) = \sigma\left(\sum_{j=0}^{M_{L-1}-1} w^L(i,j)\, f^{L-1}(j) + b^L(i)\right), \quad \text{for } i = 0,\dots,M_L - 1 \ , \tag{1.12}$$

with $M_L \in \mathbb{N}$ being equal to the number of target values $\mathbf{Y} = y(0),\dots,y(M_L-1)$, where $\mathbf{Y} \in \mathbb{R}^{M_L}$. For regression problems, a linear function is frequently chosen as the activation function in the last layer in order to cover a wide range of values. However, the choice of activation function can vary between use cases.

The most important driver for the success story of FNNs and DL, in general, is depth (Bengio et al., 2021). In 2006, Bengio et al. (2006) already describe that it is not just a matter of how many parameters an FNN has since deeper networks are inherently superior to shallower architectures with the same number of parameters, as they can generalise better. According to Bengio et al. (2021), it is more relevant that a deep FNN can compose features in the data in varying hierarchical ways at different abstraction levels. For LeCun et al. (2015), this accommodates the fact that many signals also consist of a hierarchical composition of different low-level features. However, they also note that deeper FNNs, in return, also require more data for training. Chronologically, apart from advances in network architecture, three developments have contributed significantly to making the training of deep FNNs feasible. Glorot et al. (2011) were able to show that FNNs using ReLU operations learn much faster than FNNs that rely on sigmoid functions such as the tanh, which was standard practice before. The advantage of ReLU is that its derivative is constant and, unlike tanh, does not vanish far from the zero point. The second major achievement was the use of a new regularisation technique called dropout, which greatly reduced the overfitting of deep FNNs (Hinton et al., 2012; Srivastava et al., 2014). This technique randomly freezes nodes and the corresponding connections with a specified probability so that the FNN has to distribute information more broadly across all nodes during training. Finally, Ioffe and Szegedy (2015) addressed the problem of internal covariate shifting, which led to a deceleration of the training process because the distribution of inputs of each layer always changed when the parameters of the preceding layers changed. The so-called batch normalisation keeps these distributions more stable, which allows higher learning rates to be used in the learning algorithm, speeding up the overall training process.

Convolutional Neural Networks An issue with fully-connected architectures is that they do not take into account the topology of the inputs (LeCun et al., 1999). Each value of each neuron is instead dependent on the entire input (Luo et al., 2016) and detecting the same pattern at different locations in the inputs requires that units with very similar weight distribution be spread in different locations in the NN (LeCun et al., 1999). LeCun et al. (1989) proposed a deep NN called convolutional neural network (CNN), which no longer relies on complete connections between layers but on local connections, shared weights and pooling operations, making the CNN easy to train with improved generalisation abilities. A CNN can be trained as easily as conventional FNNs by backpropagating the gradient (LeCun et al., 2015).

Given a one-dimensional input $\mathbf{X} = x(0), \ldots, x(J-1)$, the convolution in the first layer $l = 1$ of a CNN is calculated by

$$a^1(i,m) = \sum_{j=-\infty}^{\infty} w_m^1(j) \cdot x(i-j), \quad \text{for } i = 0, \ldots, N_1 - 1 \text{ and } m = 0, \ldots, M_1 - 1 \;, \tag{1.13}$$

with the weights $w_m^1 \in \mathbb{R}^{1\times k\times 1}$ referred to as kernel or filter, and the convolution output $a^1 \in \mathbb{R}^{1\times N_1\times M_1}$. The size of a^1 is determined by the number M_1 of kernels and $N_1 = J - k + 1$, where $N_1 \in \mathbb{N}$, depending on the size $k \in \mathbb{N}$ of these kernels and the input size J. The parameter k is an important parameter in a CNN because it affects the so-called receptive field of a node, which describes the size of the locally connected region. As in (1.8) for the FNN, a^1 is transformed by a non-linear activation function σ to give the activation $f^1 \in \mathbb{R}^{1\times N_1\times M_1}$ also referred to as feature map in the context of CNNs. In the following layers $l = 2, \ldots, L$, the feature map $f^{l-1} \in \mathbb{R}^{1\times N_{l-1}\times M_{l-1}}$, with $N_l = N_{l-1} - k + 1$, is convoluted with a new set M_l of kernels $w_m^l \in \mathbb{R}^{1\times k\times M_{l-1}}$ to calculate $a^l \in \mathbb{R}^{1\times N_l\times M_l}$ with

$$a^l(i,m) = \sum_{j=-\infty}^{\infty} \sum_{p=0}^{M_{l-1}-1} w_m^l(j,p) \cdot f^{l-1}(i-j,p), \quad \text{for } i = 0, \ldots, N_l - 1 \text{ and } m = 0, \ldots, M_l - 1 \tag{1.14}$$

and according to (1.8) the feature map $f^l \in \mathbb{R}^{1\times N_l\times M_l}$. For inputs with higher dimensions, such as images, (1.13) and (1.14) can be extended accordingly. The tail of a CNN usually consists of fully-connected layers, as introduced in (1.11) for FNNs, where the inputs come from the flattened feature maps of the last convolutional layer.

Similarly, as the level of abstraction increases with each layer of the NN, so does the receptive field. However, as Luo et al. (2016) show, the receptive field is usually distributed Gaussian. Thus its effective size is much smaller, so the receptive field cannot be efficiently increased by stacking only convolutional layers. Therefore, in common CNN architectures, convolutional layers, which are responsible for recognising finer and similar patterns, alternate with pooling layers, which leads to spatial invariance by reducing the resolution of the feature maps (LeCun et al., 2015). A pooling layer aggregates neighbouring points in the feature map to a mean or maximum, for example, and thereby increases the size of the receptive field multiplicatively (Luo et al., 2016). Results from Scherer et al. (2010) and Nagi et al. (2011) show that pooling with the maximum leads to fastest training and best invariance.

Also, for CNNs, depth is a key ingredient in achieving higher degrees of abstraction and, thus, better performance. Training such deep CNNs would not have been achievable in an acceptable time without developments in hardware, software and parallelisation of the training algorithms (LeCun et al., 2015). Thus, it was first the work of Krizhevsky et al. (2012), who were able to massively reduce error rates in image recognition through the efficient use of GPUs, ReLUs and dropout, that led to the breakthrough of CNNs. However, research by He and Sun (2015) and Srivastava et al. (2015) has revealed what is known as the degradation problem, that as the depth of an NN increases, the error first decreases but then saturates at a certain depth and increases steeply thereafter. Since the error increases on both the test and the training data, the cause is not overfitting but a general problem with training very deep NNs.

Residual Blocks and U-Net To overcome issues in training very deep NNs, so-called residual blocks, first described by He et al. (2016), are an important element in DL architectures nowadays. A residual block consists of two consecutive layers that represent an arbitrary function $H(x)$ to be learned. In addition to the normal flow of information, a residual block uses shortcut connections that skip the first layer of the block through identity mapping. This forces the regular stacked layers in the block to learn a function $H'(x) = H(x) - x$. At the end of the block, both pathways of information are added up ($H'(x) + x$) so that the original function $H(x)$ of the entire block is restored. The clever thing with residual blocks is that the NN can behave like a flatter NN via the shortcut connections while also building up additional knowledge in the skipped layers. Residual blocks neither increase the number of parameters in the NN nor the computational complexity compared to a standard CNN. He et al. (2016) show that a residual neural network (ResNet), a CNN with residual blocks, is superior to its counterpart without residual blocks, as it is easier to train and performs better.

Another modification of the CNN is the so-called U-Net, first published by Ronneberger et al. (2015). A U-Net consists of a so-called contracting and symmetric expanding path so that the architecture forms a U-shape. Traditional CNNs encounter the problem that they are very good at fusing information and context but thereby lose precise information about the location, which can be spatial or temporal, depending on the application. To counteract this, the U-Net first learns the context in the contracting path and localises it later in the expanding path. The contracting path follows straightforward CNN's usual structure of convolutional layers and pooling operations. On the other hand, the expanding path uses convolutions together with upsampling operations, which are exactly the inverse of a pooling operation, and very long shortcut connections. In the U-Net, however, these shortcuts do not skip single layers but enable information to travel from the very first layers to the very last layers in the NN, so that context and location can be linked again. There are some variants, such as the U-Net++ (Zhou et al., 2018) and the U-Net3+ (Huang et al., 2020), which use so-called nested and dense skip connections (U-Net++) or full-scale skip connections (U-Net3+) to reinforce the initial idea of the U-Net to localise context as precise as possible. While in the original U-Net information can only travel at the same depth within the network by a skip connection, the variants allow a more variable propagation of information.

Recurrent Neural Networks For applications with sequential inputs, such as natural language processing, recurrent neural networks (RNNs) are frequently used. RNNs sequentially process individual elements of an input sequence while storing relevant information of the past time step in a state vector (LeCun et al., 2015).

Given a multivariable time series input $\mathbf{X_t} = x_t(0), \ldots, x_t(J-1)$ at time step $t \in \mathbb{N}$, a standard RNN with $M \in \mathbb{N}$ hidden nodes calculates a time-dependent hidden state vector $h_t \in \mathbb{R}^M$ as function of the input and the previous hidden state h_{t-1} as follows

$$h_t(i) = \sigma_h \left(\sum_{j=0}^{J-1} w_{hx}(i,j) \cdot x_t(j) + \sum_{p=0}^{M-1} w_{hh}(i,p) \cdot h_{t-1}(p) + b_h(i) \right), \quad \text{for } i = 0, \ldots, M-1 \; , \tag{1.15}$$

with $w_{hx} \in \mathbb{R}^{M \times J}$ being the conventional weights between the input and hidden layer, $w_{hh} \in \mathbb{R}^{M \times M}$ representing the weights between the hidden layer state h_t at the current time step and its state $h_{t-1} \in \mathbb{R}^M$ at the preceding time step, and $b_h \in \mathbb{R}^M$ being the bias of the hidden layer. The output vector $\mathbf{Y_t} = y_t(0), \ldots, y_t(N-1)$, where $\mathbf{Y_t} \in \mathbb{R}^N$ and $N \in \mathbb{N}$, is then calculated by

$$y_t(i) = \sigma_y \left(\sum_{p=0}^{M-1} w_{yh}(i,p) \cdot h_t(p) + b_y(i) \right), \quad \text{for } i = 0, \ldots, N-1 \; , \tag{1.16}$$

with $w_{yh} \in \mathbb{R}^{N \times M}$ being the weights between the hidden layer and outputs, and the bias of the output $b_y \in \mathbb{R}^N$. The hidden state vector is always a function of all preceding time steps

$$
\begin{aligned}
h_t(i) =& g\Big(x_t(j), h_{t-1}(i)\Big) \\
=& g\Big(x_t(j), g\Big(x_{t-1}(j), h_{t-2}(i)\Big)\Big) \\
=& u\Big(x_t(j), x_{t-1}(j), x_{t-2}(j), \dots\Big), \quad \text{for } i = 0, \dots, M-1 \text{ and } j = 0, \dots, J-1 \,,
\end{aligned}
\tag{1.17}
$$

indicated by arbitrary functions g and u. Therefore, when RNNs are unfolded along time, they can be viewed as very deep FNNs that feature the same weights in each layer (Bengio et al., 1994). However, recursion causes the gradient in backpropagation to either decrease or increase at each time step, which can cause the gradient to dissipate or explode very quickly when backpropagated over many time steps (Bengio et al., 1994). Pascanu et al. (2013) were able to counteract this by applying gradient clipping when the gradient is exploding and by using a regularisation that forces the backpropagation signal not to disappear when the gradient is vanishing. However, early theoretical and empirical studies have shown that RNNs in their original form fail to store information over a longer period of time (Bengio et al., 1994).

Improved architectures such as the hierarchical RNNs by Hihi and Bengio (1995), which use multiple levels of the internal state vector that operate on different time scales, as well as RNNs with a so-called gated mechanism, which enables the RNN to decide by situation whether to retain or replenish its memory, such as the long short-term memory (LSTM) cells by Hochreiter and Schmidhuber (1997) or the gated recurrent units (GRUs) by Cho et al. (2014), allow longer-term dependencies to be captured with RNNs. Chung et al. (2014) show that, in particular, RNNs with such a gating mechanism are superior to conventional RNNs. However, they cannot make a clear conclusion as to whether LSTM or GRU perform better, which later studies by Greff et al. (2017) and Cahuantzi et al. (2021) also state. Yet, Cahuantzi et al. (2021) can identify a tendency that GRUs perform better in time series with lower complexity, while LSTMs deliver better results for more complex time series. However, the research by Zhao et al. (2020) suggests that even GRU and LSTM do not have long-term memory from a statistical perspective, and further development is needed to account for long-term dependencies.

1.3.2. Deep Learning Methods for Time Series Applications

When applying DL to time series, RNNs seem to be the most suitable means at first glance. However, Bai et al. (2018) show that simple CNNs can outperform RNNs with LSTM cells in various application tasks and show a longer memory. According to Gehring et al. (2017), the hierarchical structure of CNNs is responsible for better long-term memory, as it allows closer elements in shallow layers and more distant elements in deeper layers to interact with each other. This results in a relatively shorter learning path for the long-term dependencies than in the chained structures of an RNN. Borovykh et al. (2018) also conclude from their research that CNNs are very well suited for time series regression problems because they are much more time-efficient, easier to train and perform better. This is particularly due to the fact that CNNs can be parallelised much better so that the full capabilities of GPU hardware and software can be better exploited (Gehring et al., 2017).

However, CNNs may also have difficulties with time series, as the relevant patterns in time series are overlaid by other patterns and noise, so a feature representation on different time scales is necessary (Cui et al., 2016). Also, causality is an issue in time series, as an output at time t may only ever be dependent on time points up to and including time $t_i \leq t$ (Bai et al., 2018). Therefore, various approaches to better adapt the conventional CNN architectures to time series exist. According to Bai et al. (2018), one naïve possibility is to use causal convolutions, in which each activation at a time t is calculated only by convolving values of the current and preceding time steps rather than being centred around t. Unfortunately, causal convolutions require a very deep NN or very large filters to span a larger receptive field and capture long-term correlations. Therefore, van den Oord et al. (2016) use dilated causal convolutions, which allow a very large receptive field that increases exponentially (Borovykh et al., 2018). Dilated convolutions, as proposed by Yu and Koltun (2016), consist of the kernel running over the inputs or feature maps with larger step sizes, whereby the step size increases with increasing depth of the layer in the CNN. In the special case with a constant step size of 1, dilated convolutions result in the conventional convolution, as shown in (1.14). The results of Bai et al. (2018) and Borovykh et al. (2018) using dilated causal convolutions on time series show that they tend to be superior to the usual RNN approaches.

Another approach to handle the superposition of time scales better is multi-scale NNs that consist of several branches that extract the important signals from time series on different time scales (Cui et al., 2016). Jiang et al. (2019), for example, smooth the input signal with moving averages to different degrees and thus generate different variants of the signal, which are fed into separate branches of the NN. Cui et al. (2016) additionally use branches in the NN that sample the values of the time series with different step sizes to filter out signals. In both examples, the NN first learns local features in the branches, which are merged at deeper layers by the architecture of the NN. Finally, the use of inception blocks (Szegedy et al., 2015), which are essentially concurrent convolutions with different kernel sizes in the same layer, to extract features of different sizes and with respect to time series on different time scales, is becoming an attractive method. For example, an application of inception blocks for time series can be found in Ismail Fawaz et al. (2020) and Kleinert et al. (2021).

1.3.3. Local Ozone Prediction with Deep Learning

When starting research for this thesis in 2019, there were few publications on the prediction of ozone based on DL methods. A couple of pioneering papers appeared around the turn of the millennium. All approaches are based on simple FNNs with a single hidden layer consisting of between 4 and 10 nodes. Comrie (1997), for example, predicts ozone with such a small FNN at eight different AQSs over a period of five summer periods with higher accuracy than regression approaches, and Gardner and Dorling (2001) also use a small FNN to make predictions for ozone at six AQSs over a period of 12 years. In both studies, a separate FNN is trained for each AQS so that no statement can be made about general performance. Besides the small FNNs, it is also typical for this time period that publications are only based on very small data sets. For example, Cobourn et al. (2000) and Prybutok et al. (2000) each use only a single AQS for their studies, whereby Cobourn et al. (2000) use about 1000 samples, and Prybutok et al. (2000) use less than 150 samples. However, the research presented here as examples must be seen in the context of the progress of NN and DL research at that time. The major milestones significantly contributing to the breakthrough of DL (see Section 1.3) were not yet reached, and neither large data sets nor the required hardware and software were available at the time.

In the following years, publications were limited to the same approaches with small FNNs and small amounts of data, although DL evolved at the same time and larger and larger data sets became available. From 2017 onwards, the first larger FNNs can be found in ozone forecast applications using larger data sets. In Ghoneim et al. (2017), a deep FNN with 10 hidden layers, each consisting of 120 neurons, is used at over 400 measurement sites. However, the data set only spans three months, and all monitoring sites are installed in the same city, so the representativeness of the results should be treated with at least caution. Di et al. (2017b) use an FNN with two hidden layers of 15 nodes each on over 1800 AQSs and 13 years of data. To capture geographical characteristics, the authors use a kind of convolutional layer, where the weights are determined by the inverse distance between the AQS and geographical location rather than being trainable as in a CNN. Another use of an FNN can be found in Seltzer et al. (2020), who use three hidden layers with 32 nodes each and a data set of about 3500 pseudo AQSs from model grid points covering 16 years.

In the same period, research based on RNNs and CNNs began to appear. Navares and Aznarte (2020) follow a recurrent approach by using an RNN with 500 LSTM cells organised in a single hidden layer followed by a fully connected layer with 100 neurons. By contrast, Ma et al. (2020) use a thinner but deeper RNN consisting of 7 hidden layers with 128 LSTM units each. In Wang et al. (2020), the RNN approach (LSTM cells) is combined with two FNNs as a hybrid model, where the FNNs are responsible for processing spatial data and combining temporal and spatial information. The RNN consists of 6 layers with 256 LSTM cells each, and the two FNNs are each composed of two hidden layers with 256 neurons. Application of CNNs can be found in Eslami et al. (2020) and Sayeed et al. (2020), who each use a CNN with 5 layers. Thereby, the convolutions are applied along the temporal dimension. The results of both studies show that CNNs are superior to alternative approaches such as RNN, FNN or regression methods.

The publications listed in this section represent only a selection of the published literature on ozone prediction using DL. In spite of the general progress in DL methods described above, many studies, even after 2012, still employed methods that were in the state of research around 2000. These publications are therefore not discussed in more detail here due to lack of quality of the results, too short time series, a limited number of AQSs, a small size or outdated architecture of NNs, or missing statistical evaluation. Moreover, even higher-quality publications such as Eslami et al. (2020) and Sayeed et al. (2020) lack important information such as comparison with simple statistical methods such as persistence or climatology or with advanced models for air quality prediction such as state-of-the-art CTMs, so that a thorough evaluation of the results is not fully possible. However, incomplete presentation or lack of adequate cross-referencing of results is not only a problem in air quality research but a very serious issue across ML applications (see following Section 1.3.4). It should be noted, nevertheless, that awareness of the need for a clear presentation of results obtained with DL has steadily increased in recent years. With the pulse of time that DL is becoming more and more established and in which this thesis also belongs, an indication can be given that DL approaches are becoming more mature, especially in recent years, and that the number of high-quality publications is growing. An overview of the current research and its relevance to this thesis can be found in Section 5, as it was only published during the course of this thesis.

1.3.4. Reproducibility in Science and Deep Learning Research

In 2016, Baker (2016) reported that more than 70% of over 1500 researchers from different scientific fields participating in a survey of the Nature journal had tried and been unable to reproduce another scientist's results. More than half had even experienced a failure to reproduce their own results, raising the question of whether science is in a crisis of reproducibility. Moreover, Serra-Garcia and Gneezy (2021) find a tendency in science for publications that are not reproducible to be cited more often than reproducible research results. The authors suspect that reviewers, in particular, have to make a compromise and lower the requirements if the results are interesting enough. Likewise, in AI research, experiments and results are not sufficiently documented, as Gundersen and Kjensmo (2018) discover. They draw on studies of 400 randomly selected research publications at two major AI conference series, the International Joint Conference on AI (IJCAI, 2013 and 2016) and the Association for the Advancement of AI (AAAI, 2014 and 2016). Tatman et al. (2018) show a similar issue when analysing 679 papers presented at the 2017 Neural Information Processing Systems (NIPS) conference, where less than 40% of the papers provided links to the code. The shortcomings do not appear to be the fault of the scientists alone but, instead, arise from a counter-optimal combination of the hype about AI and a lack of effective checks and balances (Gibney, 2022). Due to the popularity and the large number of online tutorials, it is very easy to learn AI methods in a few hours (Gibney, 2022), but this does not mean that a basic understanding of, for example, statistical analysis methods or technical aspects can be acquired in the same time. According to Pineau et al. (2021), this is particularly noticeable because, in contrast to earlier years when knowledge in computer science was usually based on mathematical and theoretical investigations, new knowledge is generated through experimental work in modern times. Further, there is a tendency for publications to be biased towards exclusively positive research results rather than failure stories (Pineau et al., 2021).

Wherein mathematical equations and laws can underline the validity of results, experimental work must be reproducible to be validated in order to contribute to scientific progress (Pineau et al., 2021). However, without a more precise standardisation of what reproducibility for ML should look like, it is difficult to assess to what extent new research results actually represent an improvement on existing approaches and thus contribute to scientific progress (Henderson et al., 2018). They define *reproducibility* as the ability of an independent research team to consistently reproduce the same results using the same AI methods using documentation from the original research team. Yet, reproducibility is influenced by extrinsic factors such as hyperparameters and intrinsic factors such as the random seed of a learning algorithm, so Henderson et al. (2018) suggest that all hyperparameters, implementation details, experimental setup, and evaluation methods of the baselines and new methods are accurately described. For example, a categorisation, according to Gundersen (2021), into the four reproducibility types of general description, code, data and experiment may help. For Kapoor and Narayanan (2022), in addition, communication of the limitations and weaknesses of a novel method, as well as a somewhat sceptical attitude towards results, is part of improving reproducibility. Otherwise, Henderson et al. (2018) see an inevitable delay in progress as reported results have to be laboriously reproduced. According to Gundersen (2019), the key factors that prevent researchers from putting more effort into reproducibility are, firstly, a high time investment with no immediate results, secondly, a lack of incentives, for example, from publishers or grant makers, and thirdly, the risk of future research by sharing data and code when other research teams can quickly build on published results. For Kapoor and Narayanan (2022), there is evidence that ML-based science is in the midst of a reproducibility crisis for two major reasons. First, publications using ML methods are riddled with the same recurring pitfalls, and these spread to any field that begins to adopt ML. Second, there are no systematic solutions to prevent these avoidable errors. However, on a positive note, Gundersen and Kjensmo (2018) and Gundersen (2019) see that interest in reproducibility has increased over the years from 2012 to 2016. Thereby, adopting open science practices is an important means to strengthen reproducibility (Munafò et al., 2017), and the use of robust experimental workflows promotes reproducibility and may avoid accidental sources of error as a side effect (Pineau et al., 2021). Kapoor and Narayanan (2022) also advocate a fundamental change in the way ML is published so that sources of inconsistencies are identified before papers are submitted or, at the latest, during peer review.

The quality of research, in particular, benefits from the publication of reproducible results, as inadequate approaches can be quickly identified. This applies particularly to evaluating results since exaggerated test results can mislead people into thinking that an AI method recognised and understood the problem (Gibney, 2022). A study by Liu et al. (2019) in the field of medical imaging revealed, for example, that only 5% of 20,000 publications compared their results with health-care professionals in real-world clinical settings using the same data set. In ML, this phenomenon is referred to as data leakage. Data leakage generally describes spurious relationships between the independent variables and the target variable due to the methods of data collection, data selection or preprocessing strategies, which leads to an overestimated quality of ML-based results that are not tenable in a real-world setting or upon a more detailed review (Kapoor and Narayanan, 2022). Data leakage can be caused by a breach of the separation of training and test data, an inappropriate selection of features or an unrepresentative selection of the test environment regarding the scientific question. An overview of a total of eight different types of leakage is available in Kapoor and Narayanan (2022). Examples of data leakage have already been touched upon in Section 1.3.3 on DL applications for ozone prediction. An incorrect separation of training and test data can, for example, be assumed if the data set is divided randomly since temporally close samples of a time series often show a high degree of autocorrelation (as in Maleki et al., 2019). An illegitimate choice of features would be the use of NOx or other chemical substances as predictors for ozone since, due to the complex interrelationships in air chemistry, a prediction for NOx must inevitably include information on future ozone (as in Biancofiore et al., 2015). Data leakage due to an unsophisticated choice of the test environment is, for example, when a model is built with data only in the winter period, but then claims are made about its applicability in summer, or when the overall test period is very short, so that it is significantly influenced by the current atmospheric conditions (as in Abdul Aziz et al., 2019). Other examples include work that investigates at which threshold value an ML model delivers the best performance in predicting that the limit will be exceeded instead of evaluating the performance at a fixed threshold value (as in Gong and Ordieres-Meré, 2016). Thus, in order for a scientifically tenable claim based on ML to be made, it is absolutely necessary that a clear separation of training and testing is ensured, that the selection of features is justified and that the test setting is adequate for the scientific question (Kapoor and Narayanan, 2022).

1.4. Research Questions and Thesis Outline

As described so far in this chapter, it is of great importance to predict ground-level ozone as reliably as possible. However, conventional methods have drawbacks such as a high computational burden and systematic deviations from observations. Meanwhile, emerging approaches, for example from the field of DL, also have their challenges and cannot be applied straightforwardly in ozone prediction. In particular, the use of DL is hampered by the lack of standards in the scientific community on how experiments, parameters and findings should be presented, so that the reproducibility of many publications suffers. Therefore, in this thesis, I explore the application of DL methods for a four-day ozone forecast with special attention to the issue of reproducibility. A focus of this thesis is to address the variability and superposition of different time scales of ozone so that DL approaches can be supported. Finally, I also draw a comparison to conventional forecast methods to assess how far DL-based ozone forecasting has progressed. Given the purpose of this dissertation, the following research questions arise:

Q1 Can deep learning be used to provide reliable forecasts for ground-level ozone?

Q2 How is it possible to ensure reproducibility in creating a deep learning-based forecast system for ozone?

Q3 Is there a way to address the superposition of patterns on different time scales in order to improve the predictive performance of deep learning approaches?

Q4 What limits the attainable forecast quality of deep learning-driven ozone forecasts?

Q5 How do ozone forecasts with deep learning compare to forecasts of classical chemical transport models?

Q6 How should an operational data-driven air pollution forecasting system be composed?

These questions are addressed in a series of three publications that constitute the main body of this thesis. These three manuscripts are:

M1 Leufen, L. H., Kleinert, F., and Schultz, M. G., 2021: MLAir (v1.0) – a tool to enable fast and flexible machine learning on air data time series, *Geoscientific Model Development*, Copernicus Publications, **14**, 1553–1574, doi: 10.5194/gmd-14-1553-2021 .

M2 Leufen, L. H., Kleinert, F. and Schultz, M. G., 2022: Exploring decomposition of temporal patterns to facilitate learning of neural networks for ground-level daily maximum 8-hour average ozone prediction, *Environmental Data Science*, Cambridge University Press, **1**, p. e10. doi: 10.1017/eds.2022.9 .

M3 Leufen, L. H., Kleinert, F. and Schultz, M. G., 2023 (under review): O3ResNet: A deep learning based forecast system to predict local ground-level daily maximum 8-hour average ozone in rural and suburban environment, *Artificial Intelligence for the Earth Systems*, American Meteorological Society, **1**, *revised version under review* .

The remainder of this thesis is structured as follows. This introduction is complemented by three chapters, each of which can be assigned to one of the manuscripts M1 (Chapter 2), M2 (Chapter 3), and M3

(Chapter 4). Each chapter provides an introduction to the research paper at hand, as well as the main contents in terms of methodology and results. In addition, the authors' contributions are stated according to the Contributor Roles Taxonomy (CRediT) guidelines (NISO CRediT Working Group, 2022) and each role is assigned by initials in descending order of contribution level. The publications themselves are included at the end of this thesis in Appendix D.

Broadly speaking, all three manuscripts jointly lead to the answer of Q1. In M1 (Chapter 2), the focus is on answering Q2 by describing in detail a software environment for deploying DL prediction systems. M1 tackles the crisis of reproducibility by transparently storing all conceivable information from the raw data to the ML model and its trainable parameters to ready-to-use graphics. M1 thus lays the foundation for all other research questions to be investigated. M2 (Chapter 3) is then devoted to Q3 in particular, showing how DL methods can be aided in better understanding and predicting time series characterised by the superposition of different time scales. M2 is motivated by the fact that many existing DL-based forecasting systems at the beginning of my research period tend to collapse against a mean and can hardly produce better forecasts than simple climatological estimates. I show in M2 that decomposing the input time series into components on different time scales can improve the predictions of DL approaches. The findings from M2 lead directly to Q4 and Q5, as the models from M2 continue to trend towards a mean value with increasing lead time. In M3 (Chapter 4), I, therefore, investigate how a weather forecast can contribute to a successful ozone prediction based on DL. I show that the combination of time series decomposition with a forecast for meteorological variables and a sophisticated DL method lead to outperforming the Copernicus Atmosphere Monitoring Service (CAMS) regional ensemble forecast, which is state-of-the-art for air quality prediction in Europe. M3 is also a sophisticated blueprint for question Q6. Finally, in Chapter 5 I provide summary remarks of my research findings, positioning of these findings in the current academic context, further steps to bring the forecast methods developed into operation and suggestions for future research directions.

2. Developing a Standardised Deep Learning Workflow

published as

Leufen, L. H., Kleinert, F., and Schultz, M. G., 2021: MLAir (v1.0) – a tool to enable fast and flexible machine learning on air data time series, *Geoscientific Model Development*, Copernicus Publications, **14**, 1553–1574, doi: 10.5194/gmd-14-1553-2021 .

Author Contribution: *Conceptualisation (LHL), Data curation (LHL), Formal analysis (LHL, FK), Funding acquisition (MGS), Investigation (LHL, FK), Methodology (LHL, FK), Project administration (MGS), Resources (MGS), Software (LHL), Supervision (MGS), Validation (LHL), Visualization (LHL), Writing – original draft (LHL), Writing – review & editing (LHL, MGS, FK)*

2.1. Motivation

As the introduction chapter shows, when I started the work for this thesis, there were few high-quality DL applications for ozone forecasting and the reproducibility crisis extended across all scientific disciplines that make use of ML. Publications on DL for ozone prediction, lack, for example, a comprehensive reporting of all relevant parameters in order to understand or reproduce the results, or do not compare the findings with reference predictions such as persistence, climatology or state-of-the-art approaches, so that a general interpretation of the results is difficult. One reason for this can be seen in a lack of awareness of the necessity of reproducibility at that time. Another reason might result from the limited interdisciplinary exchange between the fields of meteorology and computer and ML sciences. This creates the risk that publications show deficits from the perspective of the other field of research. Moreover, due to a lack of standardisation, it is difficult to follow the way to research results. For example, the so-called Jupyter Notebooks are very popular for ML applications because they are very easy to use. Jupyter Notebook is a tool for interactive programming and computation in which a program code is divided into individual unstructured blocks. Since these blocks can be executed in any order and frequency, and all blocks share a common namespace, hidden dependencies quickly arise that are not immediately obvious and may eventually even prevent results from being reproduced at all. The development of a standardised workflow described in Leufen et al. (2021) therefore creates the basis for reliable DL research on ozone prediction to be reproducible and follows evaluation practises according to standards accepted in meteorology.

2.2. On a Deep Learning Life Cycle

Generally, the development of a DL model follows similar pathways. First of all, the objective is defined. Then the preparation of the data begins. This process, called the preprocessing step, includes tasks such as cleaning the raw data, handling missing values, for example by selecting a suitable interpolation method, transforming the data, for example by normalisation, and formatting the data into a suitable format for further use. Preprocessing also includes the separation into training and test data, so particular care must be taken to ensure no data leakage is introduced between either subset. This is followed by selecting an architecture for the NN and creating the initial model. Next, the learning algorithm and its hyperparameters have to be set. Subsequently, the training of this NN can begin, in which the preprocessed data are presented to the NN and the trainable parameters are adjusted through backpropagation of the error. During the training process, part of the data are used for the actual learning (training data) and an independent part of the data are used to monitor the learning progress (validation data). For example, the training can be terminated as soon as the error on the validation data does not improve for a specified period of time. Otherwise, the training process ends after a set number of iterations of the training data run through the NN. Since the development of a DL model constitutes searching for an optimal configuration of the hyperparameters and architecture, the training process must be repeated, with single or multiple parameters being varied in each training run. After a sufficiently large number of experiments with different configurations, an optimal configuration is selected based on the validation data. In addition to varying the parameters, this may also involve adjusting the data preparation process or the model architecture. As a final step, the test or evaluation phase takes place, in which the trained model is applied to the so far unused and therefore for the trained model unknown test data and statistical error analyses can be carried out. In order to avoid data leakage here as well, it is absolutely necessary that no optimisation has taken place on the test data, which also means that the selection of the best model cannot be based on the test data. However, it is common, for example, to train different network architectures separately and then let them compete with each other in the test phase. For time series problems in atmospheric science, it is a good practice to draw a comparison with reference predictions such as persistence in this step. If all the work steps and parameters required in the process are clearly documented, it is possible for third parties to follow the results and reproduce them.

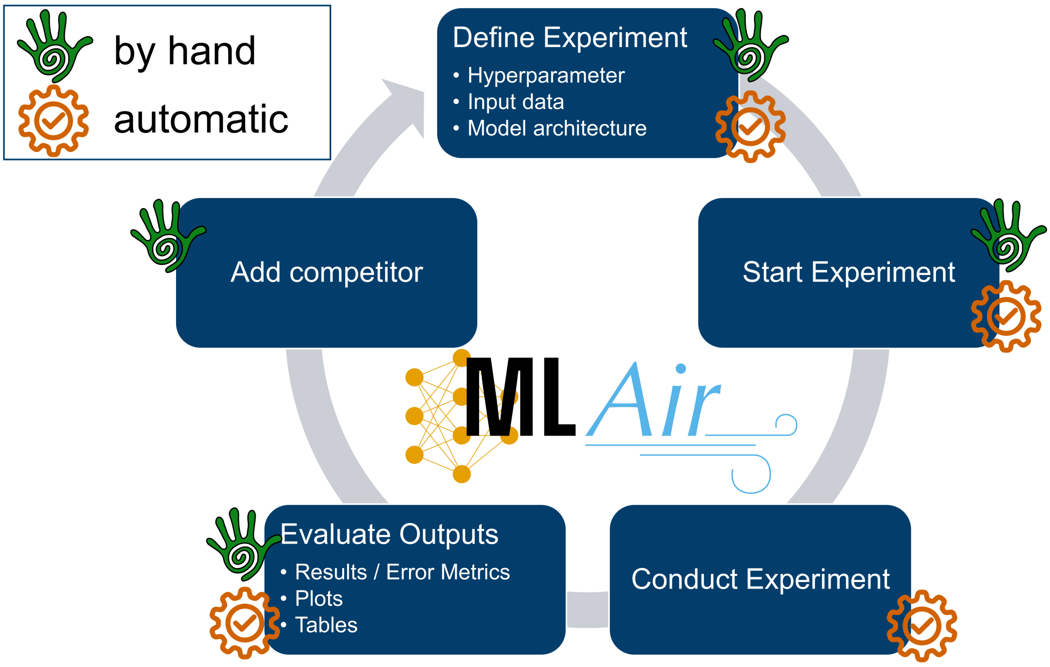


Figure 2.1.: Illustration of a typical ML workflow as it could look like when using MLAir. Starting clockwise from the top, first, the experiment is defined in a start script that contains information about hyperparameters, data preprocessing steps and the DL model architecture. Second, the experiment is started on a local machine or high performance computing (HPC) system. Third, MLAir executes a defined workflow that includes data preprocessing, model training and evaluation. Fourth, outputs such as graphics and error tables are evaluated to see the model performance. Fifth, the model can be entered into the workflow as a competitor, and then the workflow can be restarted with different parameters. Steps marked with a hand are carried out manually by the user, while those marked with a cogwheel are taken over by the MLAir software. Steps containing both icons were partially automated for the work in this thesis so that several experiment cycles for hyperparameter tuning could run automatically.

2.3. Summary of Advances

The programme Machine Learning on Air data (MLAir) described in Leufen et al. (2021) reflects exactly the DL life cycle described above. MLAir maps a single cycle from preprocessing and training up to evaluation. The search for an optimal configuration can be approached very differently, for example with a grid, random or evolutionary search strategy, and is therefore left to be done outside MLAir. In order to search for the most suitable architectures and parameters, several runs are then started according to the chosen strategy. This iterative process is illustrated in Figure 2.1. All parameters and a large amount of background information, such as the progress of the training, the number of training examples but also computed errors of the model are stored locally and are available for detailed analysis after an experiment run. The complete software code is written in the programming language Python 3, which is very flexible and has high portability on different operating systems. This makes it possible, for example, to run a large experiment on a high performance computing system but also smaller experiments on a local machine. It is also possible to repeat an experiment on another computing machine if the settings of the run are known.

By using MLAir, it is possible to strengthen reproducibility in research work. Hyperparameters, implementation details, experiment design and evaluation methods are described transparently, as required by Henderson et al. (2018), without the need for the researcher to expend much effort, as this is intrinsically built into the software. Furthermore, a meteorologically motivated evaluation strategy ensures that the evaluation of the results meets the standards of the community. Moreover, data leakage is prevented by providing a defined workflow. Finally, the work published in Leufen et al. (2021) has resulted in a number of publications. These include with Kleinert et al. (2021, 2022) and Leufen et al. (2021, 2022, under review 2023) five publications in peer-review journals as well as the two master's theses from Gramlich (2021) and Weichselbaum (2022). In addition, MLAir is being used on a pilot basis as part of the AQ Watch project (see Li et al., 2023), but related publications are still pending, and is intended to be employed in a Destination Earth use case on air quality prediction (see ECMWF, 2022).

3. Teaching Deep Learning Models to Grasp Temporal Scales

published as

Leufen, L. H., Kleinert, F. and Schultz, M. G., 2022: Exploring decomposition of temporal patterns to facilitate learning of neural networks for ground-level daily maximum 8-hour average ozone prediction, *Environmental Data Science*, Cambridge University Press, **1**, p. e10. doi: 10.1017/eds.2022.9 .

Author Contribution: *Conceptualisation (LHL, MGS, FK), Data curation (LHL, FK), Formal analysis (LHL), Funding acquisition (MGS), Investigation (LHL), Methodology (LHL), Project administration (MGS), Resources (MGS), Software (LHL, FK), Supervision (MGS), Validation (LHL, FK, MGS), Visualization (LHL, FK), Writing – original draft (LHL), Writing – review & editing (LHL, MGS, FK)*

3.1. Motivation

For the assessment of short-term ozone pollution, daily metrics such as the daily maximum 1-hour mean or the daily maximum 8-hour running average (dma8) are used (Fleming et al., 2018). The EU Ambient Air Quality Directive specifies a dma8 target value of 60 ppb for short-term ozone pollution, which must not be exceeded on more than 25 days in a year (Maas and Grennfelt, 2016). Nevertheless, the European Environment Agency (2022) estimates that 95% of the urban population in the EU is exposed to ozone concentrations above the WHO recommendations of 50 ppb (WHO, 2021). Furthermore, there are indications that exposure to ozone concentrations below the guideline values also leads to an increase in mortality, especially among vulnerable population groups (Di et al., 2017a). Therefore, a reliable prediction of dma8 ozone is essential to protect these vulnerable populations in particular, as it allows timely warnings to be issued and countermeasures to be taken.

Since ozone prediction with CTMs has shortcomings due to the high computational load and the bias against observations (see Chapter 1.1 and 1.2.3), studies such as Di et al. (2017b) and Kleinert et al. (2021) experimented with the prediction of dma8 ozone based on DL. Kleinert et al. (2021) were able to show in a study of over 300 AQSs in Germany that it is fundamentally possible to produce an ozone forecast with DL. However, the results also reveal that the DL model used has only a marginal skill compared to a linear regression approach and also cannot offer any added value for forecasts on and after the third day compared to a climatological, seasonally varying forecast, even though a deep NN with more than 300,000 weights is used with intense training. Moreover, with increasing lead time, the distribution of the forecast tends to collapse towards a monthly mean. Possible causes for this finding are the superposition of patterns on different time scales and the resulting non-stationarity, which, as shown in the introduction, is not only a general problem in the analysis of time series but can also pose challenges for DL methods in particular. Most of the training effort is spent by the DL model to learn the well-known seasonal cycle and autocorrelation, which could easily be achieved by simpler statistical approaches.

In the article Leufen et al. (2022), on which this chapter is based, I have therefore investigated how domain knowledge can help DL methods to better cope with overlapping temporal patterns so that during the learning process the focus is not on learning the obvious cycles, but on the variations from them. To showcase the improvements this approach can lead to for dma8 ozone prediction, 50 AQSs in the rural regions of the North German Plain (north of 52.5°N) are selected. This area is chosen to ensure that the AQSs are more homogeneous and that effects of local differences, e.g. due to orography, are of minor relevance. The chemical parameters of the study are taken from the Tropospheric Ozone Assessment Report database (TOAR DB, Schultz et al., 2017), with measurements originally provided by the German Environment Agency, and the meteorological variables come from the high-resolution reanalysis system COSMO-REA6 data set (REA6, Bollmeyer et al., 2015). All inputs are used in hourly resolution, the target variable dma8 ozone is in daily resolution, and the objective is to predict the dma8 ozone concentration for the following four days, hereafter referred to as D1 to D4. The forecast horizon is chosen to move beyond the persistence regime. Moreover, current air quality models as the CAMS regional ensemble (see Chapter 4) issue air quality forecasts for this horizon.

3.2. A Time-Filtered and Multi-Branch Approach

To support DL in learning different temporal scales, a decomposition of the input time series as described in Chapter 1.2.2 is utilised. This allows the NN to directly connect seasonality to the LT component without tedious training. Hence, it frees up resources for learning other relationships in the data, especially deviations from the seasonal norm. For the decomposition, an FIR filter with a Kaiser window (Kaiser, 1966) is used. When using filter methods, a distinction must be made between their application in analyses and predictions. For analyses, it is particularly important to separate the individual parts as clearly as possible. For this purpose, filters are usually centred around a point in time of interest t_i. For instance, to extract the LT component with a cutoff period of 21 days, a filter spanning 42 days and centred around t_i is needed. This, on the other hand, is not feasible when used in forecasting, since values in the future

($t_i > t_0$) may not yet be known for causality reasons. A causal application of filter methods in forecasting, applying the filter only to past values ($t_i \leq t_0$), causes phase shifts in the filtered signal and thus a delay.

To guarantee causality while at the same time achieving maximum separability, a climatological forecast is used to replace unknown future data. This climatological estimate consists of a seasonal and a diurnal variation, whereby the diurnal variation may vary according to the season. The filters are then applied to this composite time series consisting of past observations and future statistical estimates. Decomposition into four components (LT, SY, DU, ID) as in equation (1.5) and into two components (LT, ST) as in equation (1.6) are tested. The separation of the LT component is additionally tested with a cutoff period of 75 days, which, according to Rao et al. (1997), results in good separation characteristics as the correlation between the resulting components is low. In addition, I investigate whether the prediction quality improves if the DL methods also have access to the original raw data in addition to the decomposed signals. In all experiments, the input data span 65 hours from $t_0 - 64h$ to t_0, where t_0 is set to 5 p.m. local time due to the computational definition of the target variable dma8 ozone (see directive of the European Parliament and Council of the European Union, 2008). For each individual sample, the decomposition is recalculated according to a given time t_0 in order to always include the most up-to-date data without causing data leakage.

In this study, so-called multi-branch NNs are implemented. These NNs consist of several branches for the inputs, whereby each branch is fed with a decomposed component of all variables. Thus, for example, the branch with the LT components can focus on seasonality, whereas the branch using ST signals can learn short-term variability. The individual branches are combined in the NN in a deeper layer and the NN learns to weight the individual branches during the training process. In this study, multi-branch NNs based on FNN, CNN and RNN are used. First, FNNs are selected to test which decomposition can produce the best accuracy. For this purpose, for each variant of the input data decomposition, an FNN is trained by hyperparameter tuning and then compared. Second, the identified decomposition method is evaluated with the other architectures. As baselines for the analysis to see how the decomposition improves the prediction, FNN, CNN and RNN are trained on the unfiltered raw data without decomposition. Note when reading that FNNs are referred to as fully connected networks (FCNs) in this study.

3.3. Summary of Results

The evaluation of the different decomposition strategies using the multi-branch FNN shows that the best results are achieved with a decomposition of the inputs into LT and ST. The reported mean squared error (MSE) of a bootstrap analysis lies at 66 ppb^2 on average, the root mean squared error (RMSE) is therefore about 8.1 ppb. Adding raw data does not improve the result any further. By contrast, a more detailed separation of the inputs into LT, SY, DU and ID components does not provide equal benefits for the FNN at all. A possible explanation is that especially the short-term scales are difficult to separate clearly (cf. Kang et al., 2013, and Chapter 1.2.2). Furthermore, no difference in FNN performance is observed when either a cutoff of 21 or 75 days is used, while the computational effort increases in the latter case. Therefore, the subsequent analysis of the different network architectures is continued with the decomposition into LT and ST with a cutoff period of 21 days. The side-by-side comparison between the NNs trained with unfiltered data and temporally decomposed data demonstrates that an improvement in prediction quality is achieved for all three tested architectures (FNN, RNN and CNN). The largest improvement can be seen for the FNN. In absolute terms, the multi-branch NNs based on FNN and RNN provide the smallest MSE of the bootstrap analysis. Furthermore, by comparing with the DL model from Kleinert et al. (2021), a persistence forecast, and a forecast based on multiple linear regression, the multi-branch NNs of this study offer more predictive power for ozone. However, a closer look at the forecast skill as a function of the forecast horizon indicates that starting from D3, there is still a tendency for the forecasts issued to converge towards a monthly mean value, albeit less pronounced. It can therefore be concluded that the temporal decomposition of the input data contributes to an improved ozone forecast, but there are still uncertainties in the forecast and a collapse of the distribution occurs.

4. A Deep Learning Model to Forecast Ozone at Local Scale

revised version under review as

Leufen, L. H., Kleinert, F. and Schultz, M. G., 2023 (under review): O3ResNet: A deep learning based forecast system to predict local ground-level daily maximum 8-hour average ozone in rural and suburban environment, *Artificial Intelligence for the Earth Systems*, American Meteorological Society, **1**, *revised version under review* .

Author Contribution: *Conceptualisation (LHL, MGS, FK), Data curation (LHL), Formal analysis (LHL), Funding acquisition (MGS), Investigation (LHL), Methodology (LHL), Project administration (MGS), Resources (MGS), Software (LHL), Supervision (MGS), Validation (LHL), Visualization (LHL), Writing – original draft (LHL), Writing – review & editing (LHL, MGS, FK)*

4.1. Motivation

As the research results in Leufen et al. (2022) show, it is possible to improve DL-based prediction by temporal decomposition of the input data. DL models are also able to learn the relationship between meteorology and air quality. However, DL only performs well for very short lead times of up to two days and suffers for longer lead times. The NNs trained in Kleinert et al. (2021) and Leufen et al. (2022) only use past values as inputs, so the NNs have no information about future weather during the forecast period. Although the NN can support the climatological estimation about the future, such an estimation is at the same time rather insufficient, when the actual weather deviates significantly from the climatologically expected state.

In the study Leufen et al. (under review 2023) presented in this chapter, the DL approach is therefore complemented by a weather forecast, so that the uncertainty of future weather conditions in the forecast period is reduced. For this research work, the study area is extended to Central Europe, so a total of 328 AQSs are included in a period from 2000 to 2021. In total, more than 800,000 training samples, about 200,000 validation samples and 170,000 test samples are available. This big data set allows the training of advanced and deeper DL models with more layers and weights. The best DL models are then compared with forecasts of the CAMS regional ensemble (CAMS, 2020), which combines nine state-of-the-art CTMs (see Marécal et al., 2015). Chemical parameters are again taken from the TOAR DB, whereas meteorological parameters are based on the fifth generation of reanalysis data ERA5 (Hersbach et al., 2020) of the European Centre for Medium-Range Weather Forecasts (ECMWF), as REA6 or real forecast data are not available for the test period. The objective is still, as in Leufen et al. (2022) respective Chapter 3, to predict dma8 ozone for D1 to D4.

4.2. Incorporating Weather Forecasts to Reduce Uncertainty

By adding a weather forecast to the input data stream, the initial ozone forecast problem is transformed into a regression task to determine which ozone concentration can be expected under a given weather condition. Thus, uncertainty effects can be reduced. The ERA5 reanalysis data set emulates an optimal forecast since observations have already been used to adjust the reanalysis. I have used the filter approach from Leufen et al. (2022, Chapter 3) and replaced the climatological estimation with ERA5 data for all meteorological variables. In this study, the meteorological inputs are furthermore extended by the forecast horizon of the DL models to the interval $[t_0 - 3d,\ t_0 + 4d]$. For the chemical input variables, climatology is still used to avoid data leakage and also the input length remains unchanged on the interval $[t_0 - 3d,\ t_0]$. Another change is that the ST and LT components of both the meteorological and chemical variables are separately input to an independent branch, partly due to the different shapes of the inputs, but also leading to a further breakdown of information so that the multi-branch NNs in this study consist of four input branches. This enables the DL model to analyse different patterns in meteorological and chemical inputs separately at first and combine them later.

In this study, different DL approaches based on FNN, CNN and RNN as well as ResNet and U-Net architectures (see Chapter 1.3.1) are tested. In addition to investigating which DL type can produce the best forecasts, I compare the best-performing DL model to the CAMS regional ensemble. Moreover, I study the influence of the forecast horizon of the weather forecast on the DL forecast. For this purpose, the lead time of the weather forecast is gradually reduced and time behind this limit is filled with the climatological estimate. The unmodified DL model then produces a forecast based on this new input data, which is compared with the original unmodified forecast of the same model. How much the forecast quality deteriorates in the process can provide information about the influence of the weather forecast and its maximum available lead time.

4.3. Summary of Results

The analysis is partitioned into multiple steps. First, the results of the different DL architectures are compared among each other and the best-performing DL model is selected for further analysis. It can be seen that deeper and more advanced CNN architectures such as ResNet or U-Net have a smaller root mean squared error (RMSE) of 5.2 ppb on average over all forecast days compared to the FNN, CNN and RNN approaches (between 5.6 and 5.8 ppb). The predictive performance of the ResNet and the U-net are so close to each other, that the distributions of the errors over all AQSs do not differ significantly in a Mann-Whitney U test (Mann and Whitney, 1947). The best DL model is therefore selected by a bootstrap analysis in which, with a number of 1000 iterations, the errors from 36 randomly drawn monthly excerpts of all AQSs from the test data are examined. Here, the ResNet achieves the lowest error on average over all iterations, which can also be confirmed with a further Mann-Whitney U test. The DL model found in this way is referred to hereinafter as O3ResNet.

Secondly, a comparison is made between O3ResNet and the CAMS regional ensemble. This analysis shows that O3ResNet has a smaller RMSE across all AQSs and forecast days, ranging from 4.3 ppb on D1 to 5.5 ppb on D4 (5.1 ppb on average over all days). The CAMS forecast, on the other hand, has an RMSE of 7.3 ppb on D1 and 7.9 ppb on D4 (7.6 ppb on average). Looking at the mean error (ME) reveals that O3ResNet issues almost bias-free predictions, whereas CAMS forecasts suffer from a positive bias. In order to clarify how much the total RMSE is influenced by the variance or whether differences result solely from the better representation of the background concentrations, the forecasts of CAMS and O3ResNet are postprocessed. For this purpose, the overall mean value is first removed from the data for each AQS in order to examine how pronounced the effect of the bias appears overall. Through the postprocessing, the RMSE of CAMS forecasts for the D1 forecast is reduced to 6.9 ppb ($\Delta \sim$ 0.4 ppb), the improvement on D4 is similar so the RMSE lays at 7.6 ppb. Nevertheless, the bias-corrected prediction of CAMS is still outperformed by O3ResNet and it can be concluded that O3ResNet is not only better at predicting the background concentration of ozone, but also its variability. As a second test, a 30-day running mean is subtracted at each AQS as postprocessing to address seasonal effects. The prediction of CAMS in terms of RMSE can thereby be further improved and lies between 5.8 and 6.4 ppb for D1 and D4, respectively. Even when using such a kind of postprocessing, O3ResNet is still preferable to a prediction by CAMS. On top, applying the postprocessing to O3ResNet further reduces O3ResNet's RMSE to 4.1 ppb on D1 and 5.0 ppb on D4.

Thirdly, the investigation of the importance of the individual input branches shows why O3ResNet does a good job in correctly predicting the background concentration but also the variability. O3ResNet obtains most of the information from the LT components of the chemical inputs and the ST components of the meteorological parameters. The ST components of the chemicals also have an influence on the

D1 forecast as well. Hereby, the LT part of the chemical observations serves O3ResNet as a kind of bias correction, as the model can directly recognise a correct ozone level for a given AQS on the basis of past observations. Fine-tuning can also be made on D1 through the ST component, which can possibly capture the autocorrelation of ozone. This relevance naturally decreases with increasing lead time, as the autocorrelation also diminishes. The ST fraction of meteorology represents the deviation from a long-term normal state, meaning in other words, the current weather. O3ResNet is thus able to learn background concentration, the weather pattern and an autocorrelation factor and incorporate the information into the forecast.

Lastly, the dependency of O3ResNet on the lead time of the weather forecast is investigated. Since ERA5 data are, as already mentioned, a reanalysis and not an operational forecast, the investigation of the lead time can give first hints about how O3ResNet could perform in a real-time forecast environment. The first finding is that O3ResNet only needs a weather forecast of four days and that there are no spurious links to weather events in the more distant future. The second implication from the lead time investigation is that O3ResNet can understand the temporal context in the input data, as for example a 24-hour weather forecast is already sufficient for the forecast on D1 to reach original performance. Indeed, the comparison with the CAMS forecasts shows that with the help of such a 24-hour weather forecast, O3ResNet has an equal skill for D2, while the CAMS forecast itself requires a 48-hour weather forecast. This phenomenon can be observed for all forecast days.

5. Conclusions and Outlook

Accurate prediction of ground-level ozone is not only scientifically valuable but also relevant in light of the short-term and long-term burden on humankind and nature caused by direct exposure to the pollutant ozone. However, conventional prediction methods have several shortcomings. CTMs, for example, have systematic biases against observations and demand a large computational workload. Regression approaches, on the other hand, struggle with the highly non-linear relationships between weather and air quality. The availability of large data sets of in-situ observations at AQS makes it possible to use DL methods to predict ground-level ozone. This work is dedicated to predicting ozone with deep NNs based on big data. When I started this work, the ML and DL methods used for ozone forecasting either were limited to simpler methods or were only developed and evaluated on a size-limited dataset. In addition, many scientific papers exhibited deficits in clearly presenting the approach and the evaluation, making it impossible for independent research teams to reproduce the findings.

In this thesis, I have therefore focused on two important aspects. First, I investigated how creating a reliable ozone forecast using DL is possible. Second, I paid special attention to the requirements to ensure this research work remains reproducible and thus can be verified by independent scientists. Reproducibility leads to more trust in the DL methods that have been developed. To achieve these objectives, in this thesis, I first designed a standardised workflow for training DL, which formed the basis for the subsequent research to be reproducible. I then explored how a DL model can be better skilled in the training process to deal with the superposition of different temporal patterns, which is generally a major challenge for statistical methods when applied to atmospheric time series. Furthermore, I investigated to what extent DL models fail to learn the relationship between weather and air quality or whether uncertainties in the forecast result more from the inherent uncertainty about future weather events. Finally, in this thesis, I examined the advantage of deeper and more sophisticated DL architectures over simpler DL methods. The findings of this work are assessed in the context of the current scientific state-of-the-art by benchmarking against an ensemble of nine cutting-edge CTMs.

Temporal Decomposition Preceding studies by Kleinert et al. (2021) and non-meteorological ML studies such as Cui et al. (2016) show that NNs have limitations when applied to time series, as overlapping patterns are present. As a result, DL methods tend to collapse against an average. This is naturally caused since DL models are usually trained using the MSE as a loss function, and predicting the mean is a smart, albeit simplistic, strategy for optimisation. However, comparable accuracy can also be achieved with simpler statistical methods that require no expensive training, so the use of DL has no real advantage over the simpler methods. The answer to this challenge lies in a meteorologically motivated decomposition of the input data into LT and ST components separating the seasonal cycle and short-term patterns. This allows the DL model to focus immediately on the short-term patterns during training. Separating temporal patterns in a time series is particularly challenging in a forecasting setting, as data leakage must be prevented to ensure causality. In the absence of a forecast on future values, only past information may be used for the decomposition. As I show in Chapter 3, using a climatological estimate of the future is a powerful solution. Combining actual observations and climatology allows a time series to be split into different components without violating any causality constraints. Since atmospheric fluctuations are not limited to single frequencies but occur and superimpose variably, I tested different degrees of decomposition. A finer separation, where the ST components are split down further, tends to have a detrimental effect on the prediction. Similar findings were also obtained by Kang et al. (2013), who were only able to achieve a clear partitioning of the signals for the separation into LT and ST. Thanks to the application of this special preprocessing, I was able to achieve a significant improvement in the prediction of ground-level ozone, even with comparatively simple DL model architectures. The trained DL models outperform reference forecasts based on climatology, persistence, and the DL model from Kleinert et al. (2021). Regardless of the chosen architecture of the NNs, an improvement is evident for all tested models, namely FNN, RNN and CNN. Finally, however, the results also suggest that a DL approach based purely on the decomposition of the input time series also yields drawbacks as the forecast horizon increases since the climatology is not fully suited to situations of large deviations from climatological norm states.

Weather Forecast Limitations in the predictive capabilities of DL methods are not only due to the overlapping of different temporal patterns but must also be located in the complex relationship between weather and air quality. Thereby, the challenge is not in learning the general link between weather and atmospheric chemistry, but it results from the lack of insight into the future weather situation. Thus, more than knowing about past weather patterns is required to make a reliable prediction. Any information about the future contributes added value. My investigations summarised in Chapter 4 confirm this statement, as using a weather forecast leads to a substantial increase in the prediction quality of the DL-based ozone forecast. This can be observed independent of the actual architecture, as the predictions of all tested DL approaches, in this case, FNN, RNN as well as different advanced types of CNN, are significantly improved relative to the DL methods that have only access to past observations and a climatological estimate. Indeed, by fine-tuning the network architecture and hyperparameters and using large-scale data, deeper and more advanced DL approaches can be trained with even better performance. My analysis shows that DL models based on CNN with residual blocks as well as U-Nets are superior to other approaches, whereby both perform with comparable accuracy. A survey of the current literature shows that the use of U-Nets, in particular, is widespread. Sayeed et al. (2022) use a U-net for data imputation to transform in-situ observations into a spatially uniform grid, and He et al. (2022) use a U-net with additional built-in LSTM cells and over 62 million trainable parameters for a grid prediction of ozone.

Benchmarking against CTM The unique element of this work is that the results are compared with an ensemble of state-of-the-art CTMs. Few studies in ozone prediction can be found that take this step. One example is Cheng et al. (2022), who compare their DL model with the Nested Air Quality Prediction Modeling System (NAQPMS, Kong et al., 2021). However, the air quality model is superior to the DL approach at grid prediction, and the DL method only slightly improves prediction at AQS. However, as the evaluation of this study is limited to only two months, it is not yet known how representative these results are. By contrast, in the work for this dissertation, I use a significantly longer evaluation period of three years. Key findings from the comparison with the CAMS regional ensemble are that the DL approach has a considerably higher prediction performance, even when a bias correction is applied to the CTM forecasts afterwards, which corrects the major drawback of CTM-based predictions. Therefore, the DL approach can not only achieve a small bias compared to observations at AQS but also better represents the variability of local ozone.

Reproducible Workflow The basis of all results presented in this thesis is the development of a standardised workflow for training DL models on atmospheric time series, as described in Chapter 2. This workflow allows independent parties to run and reproduce experiments on an arbitrary computing system. An important feature of the workflow is that it was developed according to accepted standards from meteorology, statistics and ML, as well as best practices from software development, for example, through versioning and comprehensive testing so that functionality and reproducibility are always guaranteed. Designing a reproducible workflow is in the zeitgeist of the growing awareness of the need for reproducibility in science. This increase can be seen as relevant literature on reproducibility in ML has been published since 2018 and thus coincides with this thesis project. The proliferation of FAIR data, which requires that data be findable, accessible, interoperable and reusable (see Wilkinson et al., 2016), and the publication and use of benchmark datasets, as advocated by Kapoor and Narayanan (2022), are also indicative of this trend. In the atmospheric sciences, for example, the benchmark data sets WeatherBench (Rasp et al., 2020), AQ-Bench (Betancourt et al., 2021) or ClimateBench (Watson-Parris et al., 2022) have been published. Another positive development is the increasing requirements regarding data and code for authors when publishing research results. Though the Nature journal has been asking for code to be made publicly available whenever possible since 2014 (Nature, 2014), Liu and Salganik (2019) still found fault that peer-review practices often do not require authors to provide or publish their code and data in the review process. A certain paradigm shift is emerging here, as Nature, for example, now clearly stipulates in its policy that code and data must be made available (see Research, 2022), and also the Geoscientific Model Development (GMD) journal requires the provision of code and data during the review process in its guidelines (see Geoscientific Model Development, 2022). For the GMD journal, in particular, this explicitly excludes any kind of embargo, for example, in dependence on a successful publication or after a certain amount of time.

Limitations In spite of the significant advances made through this work, some limitations remain, which I would like to discuss in the following. The improvements in forecasting presented in Chapter 4 are associated mainly with the use of weather forecasts. However, the ERA5 data used are rather an idealised forecast, as they already include information about the future. Therefore, the DL model may be overconfident in the meteorological inputs. However, the studies by Bauer et al. (2015) and Haiden et al. (2022), for example, show that numerical weather prediction models can nowadays offer a very reliable forecast for up to one week in advance, so it can be assumed that the performance of the DL approach will not deteriorate to a large extent. However, a corresponding test to verify this assumption is still pending. A second limitation in applicability is that the model has only been trained in rural and suburban environments, so no claim can be made about its ability to make a reliable prediction in urban areas. Thirdly, in the development of the DL-based prediction system, a special focus was put on the tuning of the DL model, whereas the decomposition of the inputs was not investigated in the deepest detail. I have tested what difference the choice of decomposition can make, but I have not tested what influence the chosen decomposition method has. Neither other filters, such as the KZF or through Fourier analysis or based on wavelets, were tested. Thus, it cannot be answered whether a potentially more precise decomposition leads to an improvement in the prediction. Next, the work in this study is limited to a single objective variable, dma8 ozone, and a forecast horizon of four days for the prediction of ground-level ozone on AQS level. As a result, it is not possible to issue a grid ozone forecast. Instead, the DL model depends on the data availability at AQSs. Furthermore, the DL model does not only produce point forecasts, but it also has a very limited spatial view of the vicinity and does not use any metadata like land use, population density or topography. Hence, the model is not able to account for regional or global context information. Since transport is particularly relevant for air quality, neglecting the neighbourhood imposes limitations on the DL model. Also, given that the temporal context represented by the decomposition into LT and ST provides valuable information for the DL model, it can be concluded that the spatial context, or even the combination in the form of a spatiotemporal context, can also lead to further improvements in the prediction. Finally, it is evident in ML sciences and meteorology that applying probabilistic approaches and ensemble methods offers added value over deterministic predictions. This was also not explored in the context of this thesis. For use by decision-makers, for example, it would be very beneficial to gain further confidence in the DL method by indicating the model's confidence for each forecast issued.

Outlook Looking ahead, the limitations identified offer potential for future experimental and analytical research. In the future, it makes sense to set up the DL model on an even broader base by deepening the understanding of urban environmental conditions and subsequently using and testing the model in a larger area, such as Europe as a whole or on another continent. Extending the objective either to other ozone statistics, such as the daily maximum of one-hour values, or the prediction of hourly values to cover the whole diurnal cycle, as well as opening up to other air pollutants such as NO, NO2 or particulate matter are also ideas to be pursued further.

Methodologically, I see two developments in science that I consider most relevant for this work. These are, firstly, the further evolution in reproducible science and, secondly, the emergence of more sophisticated DL architectures. Regarding reproducibility, I have already discussed throughout this thesis the deficits that I personally, but also the scientific community, see in many ML publications. I, therefore, consider the proposal of implementing so-called model cards or model info sheets as a standard, which Mitchell et al. (2019) and Kapoor and Narayanan (2022) put forward, to be a very fascinating concept. The model cards proposed by Mitchell et al. (2019) focus very strongly on ML models and require, inter alia, a very precise description of the basic information about the model and what the intended use cases are, but also ethical considerations can be included in these cards. The model info sheets introduced by Kapoor and Narayanan (2022) are essentially an extended checklist in which authors of a study answer questions regarding a strict training-test separation, the justification behind chosen features and model structure, and the confirmation that the test data was drawn from a distribution of scientific interest. As Kapoor and Narayanan (2022) themselves note, of course, even a model info sheet is not the ultimate and above all foolproof solution, since claims, for example, cannot be verified without computational reproducibility, and disingenuous statements in a sheet can lead to false confidence about the results. Nevertheless, model info sheets help both reviewers and third-party researchers to better understand and verify the results of scientific research without going through all the tedious and time-consuming steps of the computational exact verification. Model info sheets promote reproducibility awareness and inspire practitioners to pay more attention to reproducibility.

The second important and prominent perspective is the progress of science in the development of new and more sophisticated DL architectures. Even if the publication of these methods dates back some years, their popularity is increasing year by year, as it still takes some time for these new technologies to become suitable for a broader user basis. Of particular relevance are variational autoencoders (VAEs, Kingma and Welling, 2014), generative adversarial networks (GANs, Goodfellow et al., 2014) and transformers (Vaswani et al., 2017). VAEs consist of an encoder and a decoder network, whereby from a statistical perspective, the functioning of a VAE is similar to inference in a latent Gaussian model in which neural networks parameterise posterior and model likelihood. A GAN consists of two models competing during the training process, where the generator tries to create authentic data, and the discriminator identifies it as generated and differing from real data. Through this zero-sum game, the data generated by the generator becomes more and more realistic. In particular, the combination of VAE and GAN in a so-called VAE-GAN (Larsen et al., 2016) offers enormous advantages, as GANs tend to generate blurred data, and the use of a VAE can prevent this. Transformers are based on the principle of self-attention modules that can weight the importance of elements in the input data through a combination of query and key vectors. These modules can be multi-headed so that each module can focus on particular information. Transformers have led to significant improvements over RNN approaches, especially in natural language processing (c.f. Conneau and Lample, 2019; Devlin et al., 2019). Also, the chatbot ChatGPT, which is currently being discussed all over, is based on a transformer architecture called GPT-3 (Brown et al., 2020).

Applications of these very sophisticated model types have also found their way into ozone forecasting, especially in the last year. In their study, Cheng et al. (2022) use a VAE-GAN to predict hourly and daily ozone across China. The results are promising, but their method still is underperforming an air quality model. Another relevant study by Hickman et al. (2022) is based on the use of temporal fusion transformers (TFTs, Lim et al., 2021), a derivative of transformers explicitly designed for use with multivariate time series in combination with time-invariant features. In their study on nearly a thousand AQSs in the United Kingdom, France and Italy, Hickman et al. (2022) used two million samples consisting of dynamic variables from meteorology and static features such as population density, altitude and station type for a four-day forecast of ozone. Their performance is comparable to the results presented in my research. The highlight of the TFT architecture is that it is designed to accommodate the heterogeneity in the different time series by splitting the data processing into local processing of specific patterns and global processing to fuse this information. In the context of the framework of this thesis, this represents a striking similarity in how the superposition of signals is approached. Applying the DL model from this thesis to different locations, like in Hickman et al. (2022), could reveal how the different approaches compare and to what extent a combination of the approaches can lead to an even more reliable ozone prediction.

Complementary to the further development and applicability of the DL model, an important step to be taken in the near future is migrating the DL model to an operationalised forecasting system, which will produce and publish real-time forecasts. This requires a great effort on the technical side, as up-to-date weather forecasts and AQS observations have to be collected and processed, and the infrastructure for publishing the forecasts has to be built. Despite the obstacles of a more technical nature, this goal represents a major milestone in DL-based air quality forecasting, as it would be the first time a forecast is available in live operation. Turning this theoretically motivated thesis into an actual application will be the foundation for future users, such as decision-makers and stakeholders, to access the forecasts and get their impression of potential use cases.

Research Questions and Conclusion As a last point, I would like to review the research questions raised in the introduction chapter (Section 1.4), which have guided me through the entire work of this thesis. Concerning question Q1, whether the use of DL in ozone forecasting is sensibly possible, I can give a clear yes as an answer. The second question Q2 about reproducibility can be answered to the extent that I have developed a standardised and reproducible workflow that has contributed to a number of scientific papers and findings. Measures to ensure reproducibility that I have taken in this work are (i) a clear separation of training and test data, for example, when computing transformation properties, estimating climatological statistics and during training, along the temporal axis to prevent data leakage, (ii) an appropriate choice of predictors that excludes invalid variables from the future, (iii) an estimation of how the lead time of the weather forecast will affect the DL performance, (iv) an evaluation against baseline methods as well as state-of-the-art CTMs, (v) following open science practices by publishing code and data, using a standardised workflow and precisely describing parameters and model architectures, and (vi) the clear communication of limitations of the chosen approaches. I addressed the superposition of temporal patterns (Q3) by decomposing the input time series so that the DL models exhibited increased training progress. At the same time, not only the superposition of temporal patterns but also the uncertainty about future weather limits the prediction quality of DL-based ozone forecasts. These limitations called for in Q4 can be countered by using a weather forecast, large data sets and deep and advanced DL architectures. This enabled to train a DL model that, when compared to an ensemble of cutting-edge CTMs as requested in Q5, can produce a more reliable forecast for ground-level ozone capable of representing both background levels and variability more accurately. For the final question Q6, I would like to refer to the previous outlook discussion, where I pointed out which milestones still need to be added for the findings of this thesis to be used in an operational air quality forecast. After all, the research for the use of DL in air quality prediction is not yet completed with this thesis, but opens up new and exciting research opportunities to be explored.

A. Bibliography

Abdul Aziz, F. A. B., N. Abd. Rahman, and J. Mohd Ali, 2019: Tropospheric Ozone Formation Estimation in Urban City, Bangi, Using Artificial Neural Network (ANN). *Computational Intelligence and Neuroscience*, **2019**, 1–10, https://doi.org/10.1155/2019/6252983, URL https://www.hindawi.com/journals/cin/2019/6252983/.

Bai, S., V. Koltun, and J. Z. Kolter, 2018: Convolutional Sequence Modeling Revisited. *ICLR 2018 Workshop Track*, International Conference on Learning Representations, Vancouver Convention Center, Vancouver, BC, Canada, URL https://openreview.net/forum?id=rk8wKk-R-.

Baker, M., 2016: 1,500 scientists lift the lid on reproducibility. *Nature*, **533 (7604)**, 452–454, https://doi.org/10.1038/533452a, URL https://www.nature.com/articles/533452a.

Baklanov, A., and Coauthors, 2014: Online coupled regional meteorology chemistry models in Europe: current status and prospects. *Atmospheric Chemistry and Physics*, **14 (1)**, 317–398, https://doi.org/10.5194/acp-14-317-2014, URL https://acp.copernicus.org/articles/14/317/2014/.

Bauer, P., A. Thorpe, and G. Brunet, 2015: The quiet revolution of numerical weather prediction. *Nature*, **525 (7567)**, 47–55, https://doi.org/10.1038/nature14956, URL http://www.nature.com/articles/nature14956.

Bell, M. L., A. Zanobetti, and F. Dominici, 2014: Who is More Affected by Ozone Pollution? A Systematic Review and Meta-Analysis. *American Journal of Epidemiology*, **180 (1)**, 15–28, https://doi.org/10.1093/aje/kwu115, URL https://academic.oup.com/aje/article-lookup/doi/10.1093/aje/kwu115.

Bengio, Y., P. Lamblin, D. Popovici, and H. Larochelle, 2006: Greedy Layer-Wise Training of Deep Networks. *Advances in Neural Information Processing Systems*, B. Schölkopf, J. Platt, and T. Hoffman, Eds., MIT Press, Vol. 19, URL https://proceedings.neurips.cc/paper/2006/file/5da713a690c067105aeb2fae32403405-Paper.pdf.

Bengio, Y., Y. Lecun, and G. Hinton, 2021: Deep learning for AI. *Communications of the ACM*, **64 (7)**, 58–65, https://doi.org/10.1145/3448250, URL https://dl.acm.org/doi/10.1145/3448250.

Bengio, Y., P. Simard, and P. Frasconi, 1994: Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, **5 (2)**, 157–166, https://doi.org/10.1109/72.279181, URL https://ieeexplore.ieee.org/document/279181/.

Bessagnet, B., and Coauthors, 2016: Presentation of the EURODELTA III intercomparison exercise – evaluation of the chemistry transport models' performance on criteria pollutants and joint analysis with meteorology. *Atmospheric Chemistry and Physics*, **16 (19)**, 12 667–12 701, https://doi.org/10.5194/acp-16-12667-2016, URL https://acp.copernicus.org/articles/16/12667/2016/.

Betancourt, C., T. Stomberg, R. Roscher, M. G. Schultz, and S. Stadtler, 2021: AQ-Bench: a benchmark dataset for machine learning on global air quality metrics. *Earth System Science Data*, **13 (6)**, 3013–3033, https://doi.org/10.5194/essd-13-3013-2021, URL https://essd.copernicus.org/articles/13/3013/2021/.

Bi, K., L. Xie, H. Zhang, X. Chen, X. Gu, and Q. Tian, 2022: Pangu-Weather: A 3D High-Resolution Model for Fast and Accurate Global Weather Forecast. *arXiv*, URL http://arxiv.org/abs/2211.02556.

Biancofiore, F., and Coauthors, 2015: Analysis of surface ozone using a recurrent neural network. *Science of The Total Environment*, **514**, 379–387, https://doi.org/10.1016/j.scitotenv.2015.01.106, URL https://linkinghub.elsevier.com/retrieve/pii/S004896971500128X.

Bollmeyer, C., and Coauthors, 2015: Towards a high-resolution regional reanalysis for the European CORDEX domain. *Quarterly Journal of the Royal Meteorological Society*, **141 (686)**, 1–15, https://doi.org/10.1002/qj.2486, URL https://onlinelibrary.wiley.com/doi/10.1002/qj.2486.

Borovykh, A., S. Bohte, and C. W. Oosterlee, 2018: Dilated convolutional neural networks for time series forecasting. *Journal of Computational Finance*, https://doi.org/10.21314/JCF.2019.358, URL https://www.risk.net/journal-of-computational-finance/6063376/dilated-convolutional-neural-networks-for-time-series-forecasting.

Bottou, L., and O. Bousquet, 2007: The Tradeoffs of Large Scale Learning. *Advances in Neural Information Processing Systems*, J. Platt, D. Koller, Y. Singer, and S. Roweis, Eds., Curran Associates, Inc., Vol. 20, URL https://proceedings.neurips.cc/paper/2007/file/0d3180d672e08b4c5312dcdafdf6ef36-Paper.pdf.

Brown, T., and Coauthors, 2020: Language Models are Few-Shot Learners. *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., Curran Associates, Inc., Vol. 33, 1877–1901, URL https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

Brunner, D., and Coauthors, 2015: Comparative analysis of meteorological performance of coupled chemistry-meteorology models in the context of AQMEII phase 2. *Atmospheric Environment*, **115**, 470–498, https://doi.org/10.1016/j.atmosenv.2014.12.032, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231014009807.

Cahuantzi, R., X. Chen, and S. Güttel, 2021: A comparison of LSTM and GRU networks for learning symbolic sequences. *arXiv*, URL http://arxiv.org/abs/2107.02248.

Camalier, L., W. Cox, and P. Dolwick, 2007: The effects of meteorology on ozone in urban areas and their use in assessing ozone trends. *Atmospheric Environment*, **41 (33)**, 7127–7137, https://doi.org/10.1016/j.atmosenv.2007.04.061, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231007004165.

CAMS, 2020: *Regional Production, Updated documentation covering all Regional operational systems and the ENSEMBLE*. Copernicus Atmosphere Monitoring Service. Regional Air Quality Production Systems, URL https://atmosphere.copernicus.eu/sites/default/files/2020-09/CAMS50_2018SC2_D2.0.2-U2_Models_documentation_202003_v2.pdf.

Cheng, C. S., and Coauthors, 2007: A Synoptic Climatological Approach to Assess Climatic Impact on Air Quality in South-central Canada. Part I: Historical Analysis. *Water, Air, and Soil Pollution*, **182 (1-4)**, 131–148, https://doi.org/10.1007/s11270-006-9327-3, URL https://link.springer.com/10.1007/s11270-006-9327-3.

Cheng, M., F. Fang, I. M. Navon, J. Zheng, X. Tang, J. Zhu, and C. Pain, 2022: Spatio-Temporal Hourly and Daily Ozone Forecasting in China Using a Hybrid Machine Learning Model: Autoencoder and Generative Adversarial Networks. *Journal of Advances in Modeling Earth Systems*, **14 (3)**, https://doi.org/10.1029/2021MS002806, URL https://onlinelibrary.wiley.com/doi/10.1029/2021MS002806.

Cho, K., B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, 2014: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *arXiv*, https://doi.org/10.48550/ARXIV.1406.1078, URL https://arxiv.org/abs/1406.1078.

Choromanska, A., M. Henaff, M. Mathieu, G. Ben Arous, and Y. LeCun, 2015: The Loss Surfaces of Multilayer Networks. *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, G. Lebanon, and S. V. N. Vishwanathan, Eds., PMLR, San Diego, California, USA, Proceedings of Machine Learning Research, Vol. 38, 192–204, URL https://proceedings.mlr.press/v38/choromanska15.html.

Chung, J., C. Gulcehre, K. Cho, and Y. Bengio, 2014: Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv*, URL http://arxiv.org/abs/1412.3555.

Cobourn, W., and M. C. Hubbard, 1999: An enhanced ozone forecasting model using air mass trajectory analysis. *Atmospheric Environment*, **33 (28)**, 4663–4674, https://doi.org/10.1016/S1352-2310(99)00240-X, URL https://linkinghub.elsevier.com/retrieve/pii/S135223109900240X.

Cobourn, W. G., L. Dolcine, M. French, and M. C. Hubbard, 2000: A Comparison of Nonlinear Regression and Neural Network Models for Ground-Level Ozone Forecasting. *Journal of the Air & Waste*

Management Association, **50 (11)**, 1999–2009, https://doi.org/10.1080/10473289.2000.10464228, URL https://www.tandfonline.com/doi/full/10.1080/10473289.2000.10464228.

Cohen, A. J., and Coauthors, 2017: Estimates and 25-year trends of the global burden of disease attributable to ambient air pollution: an analysis of data from the Global Burden of Diseases Study 2015. *The Lancet*, **389 (10082)**, 1907–1918, https://doi.org/10.1016/S0140-6736(17)30505-6, URL https://linkinghub.elsevier.com/retrieve/pii/S0140673617305056.

Colette, A., B. Bessagnet, F. Meleux, E. Terrenoire, and L. Rouïl, 2014: Frontiers in air quality modelling. *Geoscientific Model Development*, **7 (1)**, 203–210, https://doi.org/10.5194/gmd-7-203-2014, URL https://gmd.copernicus.org/articles/7/203/2014/.

Comrie, A. C., 1997: Comparing Neural Networks and Regression Models for Ozone Forecasting. *Journal of the Air & Waste Management Association*, **47 (6)**, 653–663, https://doi.org/10.1080/10473289.1997.10463925, URL https://www.tandfonline.com/doi/full/10.1080/10473289.1997.10463925.

Conneau, A., and G. Lample, 2019: Cross-lingual Language Model Pretraining. *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d. Alché-Buc, E. Fox, and R. Garnett, Eds., Curran Associates, Inc., Vol. 32, URL https://proceedings.neurips.cc/paper/2019/file/c04c19c2c2474dbf5f7ac4372c5b9af1-Paper.pdf.

Cui, Z., W. Chen, and Y. Chen, 2016: Multi-Scale Convolutional Neural Networks for Time Series Classification. *arXiv*, https://doi.org/10.48550/ARXIV.1603.06995, URL https://arxiv.org/abs/1603.06995.

Dauphin, Y. N., R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio, 2014: Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., Curran Associates, Inc., Vol. 27, URL https://proceedings.neurips.cc/paper/2014/file/17e23e50bedc63b4095e3d8204ce063b-Paper.pdf.

Demuzere, M., and N. P. van Lipzig, 2010: A new method to estimate air-quality levels using a synoptic-regression approach. Part I: Present-day O3 and PM10 analysis. *Atmospheric Environment*, **44 (10)**, 1341–1355, https://doi.org/10.1016/j.atmosenv.2009.06.029, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231009005068.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova, 2019: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186, https://doi.org/10.18653/v1/N19-1423, URL https://aclanthology.org/N19-1423.

Di, Q., L. Dai, Y. Wang, A. Zanobetti, C. Choirat, J. D. Schwartz, and F. Dominici, 2017a: Association of Short-term Exposure to Air Pollution With Mortality in Older Adults. *JAMA*, **318 (24)**, 2446, https://doi.org/10.1001/jama.2017.17923, URL http://jama.jamanetwork.com/article.aspx?doi=10.1001/jama.2017.17923.

Di, Q., S. Rowland, P. Koutrakis, and J. Schwartz, 2017b: A hybrid model for spatially and temporally resolved ozone exposures in the continental United States. *Journal of the Air & Waste Management Association*, **67 (1)**, 39–52, https://doi.org/10.1080/10962247.2016.1200159, URL https://www.tandfonline.com/doi/full/10.1080/10962247.2016.1200159.

Dueben, P. D., M. G. Schultz, M. Chantry, D. J. Gagne, D. M. Hall, and A. McGovern, 2022: Challenges and Benchmark Datasets for Machine Learning in the Atmospheric Sciences: Definition, Status, and Outlook. *Artificial Intelligence for the Earth Systems*, **1 (3)**, e210 002, https://doi.org/10.1175/AIES-D-21-0002.1, URL https://journals.ametsoc.org/view/journals/aies/1/3/AIES-D-21-0002.1.xml.

ECMWF, 2022: Exploring ECMWF's digital twins' applications for air quality analysis and forecasts. Destination Earth (DestinE). European Centre for Medium-Range Weather Forecasts, URL https://stories.ecmwf.int/exploring-ecmwfs-digital-twins-applications-for-air-quality-analysis-and-forecasts/index.html, accessed on 2023-02-20.

Eskridge, R. E., J. Y. Ku, S. T. Rao, P. S. Porter, and I. G. Zurbenko, 1997: Separating Different Scales of Motion in Time Series of Meteorological Variables. *Bulletin of the American Meteorological Society*, **78 (7)**, 1473–1483, https://doi.org/10.1175/1520-0477(1997)078<1473:SDSOMI>2.0.CO;2, URL http://journals.ametsoc.org/doi/10.1175/1520-0477(1997)078<1473:SDSOMI>2.0.CO;2.

Eslami, E., Y. Choi, Y. Lops, and A. Sayeed, 2020: A real-time hourly ozone prediction system using deep convolutional neural network. *Neural Computing and Applications*, **32 (13)**, 8783–8797, https://doi.org/10.1007/s00521-019-04282-x, URL http://link.springer.com/10.1007/s00521-019-04282-x.

European Environment Agency, 2022: *Europe's air quality status 2022*. EEA Briefing, Publications Office, LU, URL https://data.europa.eu/doi/10.2800/049755.

European Parliament, and Council of the European Union, 2008: Directive 2008/50/EC of the European Parliament and of the Council of 21 May 2008 on ambient air quality and cleaner air for Europe. *Official Journal of the European Union*, URL http://data.europa.eu/eli/dir/2008/50/oj.

Fiore, A. M., V. Naik, and E. M. Leibensperger, 2015: Air Quality and Climate Connections. *Journal of the Air & Waste Management Association*, **65 (6)**, 645–685, https://doi.org/10.1080/10962247.2015.1040526, URL https://www.tandfonline.com/doi/full/10.1080/10962247.2015.1040526.

Fleming, Z. L., and Coauthors, 2018: Tropospheric Ozone Assessment Report: Present-day ozone distribution and trends relevant to human health. *Elementa: Science of the Anthropocene*, **6**, 12, https://doi.org/10.1525/elementa.273, URL https://online.ucpress.edu/elementa/article/doi/10.1525/elementa.273/112792/Tropospheric-Ozone-Assessment-Report-Present-day.

Fuentes, M., and A. E. Raftery, 2005: Model Evaluation and Spatial Interpolation by Bayesian Combination of Observations with Outputs from Numerical Models. *Biometrics*, **61 (1)**, 36–45, https://doi.org/10.1111/j.0006-341X.2005.030821.x, URL https://onlinelibrary.wiley.com/doi/10.1111/j.0006-341X.2005.030821.x.

Galmarini, S., I. Kioutsioukis, and E. Solazzo, 2013: E pluribus unum*: ensemble air quality predictions. *Atmospheric Chemistry and Physics*, **13 (14)**, 7153–7182, https://doi.org/10.5194/acp-13-7153-2013, URL https://acp.copernicus.org/articles/13/7153/2013/.

Gao, Z., Y. Wang, P. Vasilakos, C. E. Ivey, K. Do, and A. G. Russell, 2022: Predicting peak daily maximum 8 h ozone and linkages to emissions and meteorology in Southern California using machine learning methods (SoCAB-8HR V1.0). *Geoscientific Model Development*, **15 (24)**, 9015–9029, https://doi.org/10.5194/gmd-15-9015-2022, URL https://gmd.copernicus.org/articles/15/9015/2022/.

Gardner, M., and S. Dorling, 2001: Artificial Neural Network-Derived Trends in Daily Maximum Surface Ozone Concentrations. *Journal of the Air & Waste Management Association*, **51 (8)**, 1202–1210, https://doi.org/10.1080/10473289.2001.10464338, URL https://www.tandfonline.com/doi/full/10.1080/10473289.2001.10464338.

Gehring, J., M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, 2017: Convolutional Sequence to Sequence Learning. *Proceedings of the 34th International Conference on Machine Learning*, D. Precup, and Y. W. Teh, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 70, 1243–1252, URL https://proceedings.mlr.press/v70/gehring17a.html.

Geoscientific Model Development, 2022: GMD code and data policy. Copernicus Publications, URL https://www.geoscientific-model-development.net/policies/code_and_data_policy.html, accessed on 2023-02-22.

Ghoneim, O. A., Doreswamy, and B. Manjunatha, 2017: Forecasting of ozone concentration in smart city using deep learning. *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, IEEE, Udupi, 1320–1326, https://doi.org/10.1109/ICACCI.2017.8126024, URL http://ieeexplore.ieee.org/document/8126024/.

Gibney, E., 2022: Could machine learning fuel a reproducibility crisis in science? *Nature*, **608 (7922)**, 250–251, https://doi.org/10.1038/d41586-022-02035-w, URL https://www.nature.com/articles/d41586-022-02035-w.

Glorot, X., A. Bordes, and Y. Bengio, 2011: Deep Sparse Rectifier Neural Networks. *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, G. Gordon, D. Dunson, and M. Dudík, Eds., PMLR, Fort Lauderdale, FL, USA, Proceedings of Machine Learning Research, Vol. 15, 315–323, URL https://proceedings.mlr.press/v15/glorot11a.html.

Gong, B., and J. Ordieres-Meré, 2016: Prediction of daily maximum ozone threshold exceedances by preprocessing and ensemble artificial intelligence techniques: Case study of Hong Kong. *Environmental Modelling & Software*, **84**, 290–303, https://doi.org/10.1016/j.envsoft.2016.06.020, URL https://linkinghub.elsevier.com/retrieve/pii/S1364815216302602.

Goodfellow, I., J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, 2014: Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, Eds., Curran Associates, Inc., Vol. 27, URL https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf.

Gramlich, V., 2021: Deep learning methods for forecasting of extreme ambient ozone values. Master's thesis, University of Cologne, Cologne, Germany, URL https://juser.fz-juelich.de/record/906244/files/Gramlich_Master%20Thesis%20Final%20hochgeladen%202_12_21.pdf.

Greff, K., R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, 2017: LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, **28 (10)**, 2222–2232, https://doi.org/10.1109/TNNLS.2016.2582924, URL http://ieeexplore.ieee.org/document/7508408/.

Gundersen, O. E., 2019: Standing on the Feet of Giants — Reproducibility in AI. *AI Magazine*, **40 (4)**, 9–23, https://doi.org/10.1609/aimag.v40i4.5185, URL https://ojs.aaai.org/index.php/aimagazine/article/view/5185.

Gundersen, O. E., 2021: The fundamental principles of reproducibility. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, **379 (2197)**, 20200 210, https://doi.org/10.1098/rsta.2020.0210, URL https://royalsocietypublishing.org/doi/10.1098/rsta.2020.0210.

Gundersen, O. E., and S. Kjensmo, 2018: State of the Art: Reproducibility in Artificial Intelligence. *Proceedings of the AAAI Conference on Artificial Intelligence*, **32 (1)**, https://doi.org/10.1609/aaai.v32i1.11503, URL https://ojs.aaai.org/index.php/AAAI/article/view/11503.

Guo, J. J., A. M. Fiore, L. T. Murray, D. A. Jaffe, J. L. Schnell, C. T. Moore, and G. P. Milly, 2018: Average versus high surface ozone levels over the continental USA: model bias, background influences, and interannual variability. *Atmospheric Chemistry and Physics*, **18 (16)**, 12 123–12 140, https://doi.org/10.5194/acp-18-12123-2018, URL https://acp.copernicus.org/articles/18/12123/2018/.

Haiden, T., M. Janousek, F. Vitart, Z. Ben-Bouallegue, L. Ferranti, F. Prates, and D. Richardson, 2022: Evaluation of ECMWF forecasts, including the 2021 upgrade. *European Centre for Medium-Range Weather Forecasts*, https://doi.org/10.21957/XQNU5O3P, URL https://www.ecmwf.int/node/20469.

He, K., and J. Sun, 2015: Convolutional neural networks at constrained time cost. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Boston, MA, USA, 5353–5360, https://doi.org/10.1109/CVPR.2015.7299173, URL http://ieeexplore.ieee.org/document/7299173/.

He, K., X. Zhang, S. Ren, and J. Sun, 2016: Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Las Vegas, NV, USA, 770–778, https://doi.org/10.1109/CVPR.2016.90, URL http://ieeexplore.ieee.org/document/7780459/.

He, T., and Coauthors, 2022: Deep learning to evaluate US NO_x emissions using surface ozone predictions. *Journal of Geophysical Research: Atmospheres*, https://doi.org/10.1029/2021JD035597, URL https://onlinelibrary.wiley.com/doi/10.1029/2021JD035597.

Henderson, P., R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, 2018: Deep Reinforcement Learning That Matters. *Proceedings of the AAAI Conference on Artificial Intelligence*, **32 (1)**, https://doi.org/10.1609/aaai.v32i1.11694, URL https://ojs.aaai.org/index.php/AAAI/article/view/11694.

Hersbach, H., and Coauthors, 2020: The ERA5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, **146 (730)**, 1999–2049, https://doi.org/10.1002/qj.3803, URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.3803.

Hickman, S., P. Griffiths, A. Archibald, P. Nowack, and E. Alhajjar, 2022: Forecasting European Ozone Air Pollution With Transformers. *NeurIPS 2022 Workshop on Tackling Climate Change with Machine Learning*, URL https://www.climatechange.ai/papers/neurips2022/33.

Hihi, S., and Y. Bengio, 1995: Hierarchical Recurrent Neural Networks for Long-Term Dependencies. *Advances in Neural Information Processing Systems*, D. Touretzky, M. C. Mozer, and M. Hasselmo, Eds., MIT Press, Vol. 8, URL https://proceedings.neurips.cc/paper/1995/file/c667d53acd899a97a85de0c201ba99be-Paper.pdf.

Hinton, G. E., N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, 2012: Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*, https://doi.org/10.48550/ARXIV.1207.0580, URL https://arxiv.org/abs/1207.0580.

Hochreiter, S., and J. Schmidhuber, 1997: Long Short-Term Memory. *Neural Computation*, **9 (8)**, 1735–1780, https://doi.org/10.1162/neco.1997.9.8.1735, URL https://direct.mit.edu/neco/article/9/8/1735-1780/6109.

Hogrefe, C., S. Vempaty, S. Rao, and P. Porter, 2003: A comparison of four techniques for separating different time scales in atmospheric variables. *Atmospheric Environment*, **37 (3)**, 313–325, https://doi.org/10.1016/S1352-2310(02)00897-X, URL https://linkinghub.elsevier.com/retrieve/pii/S135223100200897X.

Huang, H., and Coauthors, 2020: UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation. *arXiv*, https://doi.org/10.48550/ARXIV.2004.08790, URL https://arxiv.org/abs/2004.08790.

Im, U., and Coauthors, 2015: Evaluation of operational on-line-coupled regional air quality models over Europe and North America in the context of AQMEII phase 2. Part I: Ozone. *Atmospheric Environment*, **115**, 404–420, https://doi.org/10.1016/j.atmosenv.2014.09.042, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231014007353.

Ioffe, S., and C. Szegedy, 2015: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *Proceedings of the 32nd International Conference on Machine Learning*, F. Bach, and D. Blei, Eds., PMLR, Lille, France, Proceedings of Machine Learning Research, Vol. 37, 448–456, URL https://proceedings.mlr.press/v37/ioffe15.html.

Ismail Fawaz, H., and Coauthors, 2020: InceptionTime: Finding AlexNet for time series classification. *Data Mining and Knowledge Discovery*, **34 (6)**, 1936–1962, https://doi.org/10.1007/s10618-020-00710-y, URL https://link.springer.com/10.1007/s10618-020-00710-y.

Jahn, S., and E. Hertig, 2021: Modeling and projecting health-relevant combined ozone and temperature events in present and future Central European climate. *Air Quality, Atmosphere & Health*, **14 (4)**, 563–580, https://doi.org/10.1007/s11869-020-00961-0, URL https://link.springer.com/10.1007/s11869-020-00961-0.

Jiang, G., H. He, J. Yan, and P. Xie, 2019: Multiscale Convolutional Neural Networks for Fault Diagnosis of Wind Turbine Gearbox. *IEEE Transactions on Industrial Electronics*, **66 (4)**, 3196–3207, https://doi.org/10.1109/TIE.2018.2844805, URL https://ieeexplore.ieee.org/document/8384293/.

Junge, C. E., 1974: Residence time and variability of tropospheric trace gases. *Tellus A: Dynamic Meteorology and Oceanography*, **26 (4)**, 477, https://doi.org/10.3402/tellusa.v26i4.9853, URL https://a.tellusjournals.se/article/10.3402/tellusa.v26i4.9853/.

Kaiser, J. F., 1966: Digital filters. *System analysis by digital computer*, F. F. Kuo, and J. F. Kaiser, Eds., Wiley New York, NY, 218–285, section: 7.

Kang, D., C. Hogrefe, K. L. Foley, S. L. Napelenok, R. Mathur, and S. T. Rao, 2013: Application of the Kolmogorov–Zurbenko filter and the decoupled direct 3D method for the dynamic evaluation of a regional air quality model. *Atmospheric Environment*, **80**, 58–69, https://doi.org/10.1016/j.atmosenv.2013.04.046, URL https://www.sciencedirect.com/science/article/pii/S1352231013003002.

Kapoor, S., and A. Narayanan, 2022: Leakage and the Reproducibility Crisis in ML-based Science. *arXiv*, https://doi.org/10.48550/ARXIV.2207.07048, URL https://arxiv.org/abs/2207.07048.

Kingma, D. P., and J. Ba, 2014: Adam: A Method for Stochastic Optimization. *arXiv*, https://doi.org/10.48550/ARXIV.1412.6980, URL https://arxiv.org/abs/1412.6980.

Kingma, D. P., and M. Welling, 2014: Auto-encoding variational Bayes. *Proceedings of the International Conference on Learning Representations 2014*, Y. Bengio, and Y. LeCun, Eds., ICLR, International Conference on Learning Representations, https://doi.org/10.48550/ARXIV.1312.6114, URL https://arxiv.org/abs/1312.6114.

Kleinert, F., L. H. Leufen, A. Lupascu, T. Butler, and M. G. Schultz, 2022: Representing chemical history in ozone time-series predictions – a model experiment study building on the MLAir (v1.5) deep learning framework. *Geoscientific Model Development*, **15 (23)**, 8913–8930, https://doi.org/10.5194/gmd-15-8913-2022, URL https://gmd.copernicus.org/articles/15/8913/2022/.

Kleinert, F., L. H. Leufen, and M. G. Schultz, 2021: IntelliO3-ts v1.0: a neural network approach to predict near-surface ozone concentrations in Germany. *Geoscientific Model Development*, **14 (1)**, 1–25, https://doi.org/10.5194/gmd-14-1-2021, URL https://gmd.copernicus.org/articles/14/1/2021/, number: 1.

Kong, L., and Coauthors, 2021: A 6-year-long (2013–2018) high-resolution air quality reanalysis dataset in China based on the assimilation of surface observations from CNEMC. *Earth System Science Data*, **13 (2)**, 529–570, https://doi.org/10.5194/essd-13-529-2021, URL https://essd.copernicus.org/articles/13/529/2021/.

Krizhevsky, A., I. Sutskever, and G. E. Hinton, 2012: ImageNet Classification with Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger, Eds., Curran Associates, Inc., Vol. 25, URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Lam, R., and Coauthors, 2022: GraphCast: Learning skillful medium-range global weather forecasting. *arXiv*, URL http://arxiv.org/abs/2212.12794.

Larsen, A. B. L., S. K. Sønderby, H. Larochelle, and O. Winther, 2016: Autoencoding beyond pixels using a learned similarity metric. *Proceedings of The 33rd International Conference on Machine Learning*, M. F. Balcan, and K. Q. Weinberger, Eds., PMLR, New York, New York, USA, Proceedings of Machine Learning Research, Vol. 48, 1558–1566, URL https://proceedings.mlr.press/v48/larsen16.html.

LeCun, Y., 2019: 1.1 Deep Learning Hardware: Past, Present, and Future. *2019 IEEE International Solid- State Circuits Conference - (ISSCC)*, IEEE, San Francisco, CA, USA, 12–19, https://doi.org/10.1109/ISSCC.2019.8662396, URL https://ieeexplore.ieee.org/document/8662396/.

LeCun, Y., Y. Bengio, and G. Hinton, 2015: Deep learning. *Nature*, **521 (7553)**, 436–444, https://doi.org/10.1038/nature14539, URL http://www.nature.com/articles/nature14539.

LeCun, Y., B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, 1989: Handwritten Digit Recognition with a Back-Propagation Network. *Advances in Neural Information Processing Systems*, D. Touretzky, Ed., Morgan-Kaufmann, Vol. 2, URL https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f209556518c2fcb54-Paper.pdf.

LeCun, Y., P. Haffner, L. Bottou, and Y. Bengio, 1999: Object Recognition with Gradient-Based Learning. *Shape, Contour and Grouping in Computer Vision*, Vol. 1681, Springer Berlin Heidelberg, Berlin, Heidelberg, 319–345, https://doi.org/10.1007/3-540-46805-6_19, URL http://link.springer.com/10.1007/3-540-46805-6_19, series Title: Lecture Notes in Computer Science.

Leufen, L. H., F. Kleinert, and M. G. Schultz, 2021: MLAir (v1.0) – a tool to enable fast and flexible machine learning on air data time series. *Geoscientific Model Development*, **14 (3)**, 1553–1574, https://doi.org/10.5194/gmd-14-1553-2021, URL https://gmd.copernicus.org/articles/14/1553/2021/.

Leufen, L. H., F. Kleinert, and M. G. Schultz, 2022: Exploring decomposition of temporal patterns to facilitate learning of neural networks for ground-level daily maximum 8-hour average ozone prediction. *Environmental Data Science*, **1**, e10, https://doi.org/10.1017/eds.2022.9, URL https://www.cambridge.org/core/product/identifier/S2634460222000097/type/journal_article, publisher: Cambridge University Press.

Leufen, L. H., F. Kleinert, and M. G. Schultz, under review 2023: O3ResNet: A deep learning based forecast system to predict local ground-level daily maximum 8-hour average ozone in rural and suburban environment. *Artificial Intelligence for the Earth Systems*, **1**, publisher: American Meteorological Society.

Levy, H., 1971: Normal Atmosphere: Large Radical and Formaldehyde Concentrations Predicted. *Science*, **173 (3992)**, 141–143, https://doi.org/10.1126/science.173.3992.141, URL https://www.science.org/doi/10.1126/science.173.3992.141.

Li, C. W. Y., and Coauthors, 2023: Introduction to the AQ-WATCH multi-model air quality forecast system. *EGU General Assembly Conference Abstracts*, Vienna, EGU23–15 547, EGU General Assembly Conference Abstracts, https://doi.org/10.5194/egusphere-egu23-15547.

Lim, B., S. \. Arık, N. Loeff, and T. Pfister, 2021: Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, **37 (4)**, 1748–

1764, https://doi.org/10.1016/j.ijforecast.2021.03.012, URL https://linkinghub.elsevier.com/retrieve/pii/S0169207021000637.

Liu, D. M., and M. J. Salganik, 2019: Successes and Struggles with Computational Reproducibility: Lessons from the Fragile Families Challenge. *Socius: Sociological Research for a Dynamic World*, **5**, 237802311984 980, https://doi.org/10.1177/2378023119849803, URL http://journals.sagepub.com/doi/10.1177/2378023119849803.

Liu, X., and Coauthors, 2019: A comparison of deep learning performance against health-care professionals in detecting diseases from medical imaging: a systematic review and meta-analysis. *The Lancet Digital Health*, **1 (6)**, e271–e297, https://doi.org/10.1016/S2589-7500(19)30123-2, URL https://linkinghub.elsevier.com/retrieve/pii/S2589750019301232.

Luo, W., Y. Li, R. Urtasun, and R. Zemel, 2016: Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. *Advances in Neural Information Processing Systems*, D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, Eds., Curran Associates, Inc., Vol. 29, URL https://proceedings.neurips.cc/paper/2016/file/c8067ad1937f728f51288b3eb986afaa-Paper.pdf.

Ma, J., Z. Li, J. C. Cheng, Y. Ding, C. Lin, and Z. Xu, 2020: Air quality prediction at new stations using spatially transferred bi-directional long short-term memory network. *Science of The Total Environment*, **705**, 135 771, https://doi.org/10.1016/j.scitotenv.2019.135771, URL https://linkinghub.elsevier.com/retrieve/pii/S0048969719357663.

Maas, R., and P. Grennfelt, Eds., 2016: *Towards cleaner air. Scientific Assessment Report 2016*. EMEP Steering Body and Working Group on Effects of the UNECE Air Convention, Oslo, URL https://unece.org/sites/default/files/2021-06/CLRTAP_Scientific_Assessment_Report_en.pdf.

Maleki, H., A. Sorooshian, G. Goudarzi, Z. Baboli, Y. Tahmasebi Birgani, and M. Rahmati, 2019: Air pollution prediction by using an artificial neural network model. *Clean Technologies and Environmental Policy*, **21 (6)**, 1341–1352, https://doi.org/10.1007/s10098-019-01709-w, URL http://link.springer.com/10.1007/s10098-019-01709-w.

Manders, A. M. M., E. van Meijgaard, A. C. Mues, R. Kranenburg, L. H. van Ulft, and M. Schaap, 2012: The impact of differences in large-scale circulation output from climate models on the regional modeling of ozone and PM. *Atmospheric Chemistry and Physics*, **12 (20)**, 9441–9458, https://doi.org/10.5194/acp-12-9441-2012, URL https://acp.copernicus.org/articles/12/9441/2012/.

Mann, H. B., and D. R. Whitney, 1947: On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, **18 (1)**, 50–60, URL http://www.jstor.org/stable/2236101, publisher: Institute of Mathematical Statistics.

Marécal, V., and Coauthors, 2015: A regional air quality forecasting system over Europe: the MACC-II daily ensemble production. *Geoscientific Model Development*, **8 (9)**, 2777–2813, https://doi.org/10.5194/gmd-8-2777-2015, URL https://gmd.copernicus.org/articles/8/2777/2015/.

Meyer, P. G., H. Kantz, and Y. Zhou, 2021: Characterizing variability and predictability for air pollutants with stochastic models. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, **31 (3)**, 033 148, https://doi.org/10.1063/5.0041120, URL https://aip.scitation.org/doi/10.1063/5.0041120.

Mills, G., and Coauthors, 2018: Tropospheric Ozone Assessment Report: Present-day tropospheric ozone distribution and trends relevant to vegetation. *Elementa: Science of the Anthropocene*, **6**, 47, https://doi.org/10.1525/elementa.302, URL https://online.ucpress.edu/elementa/article/doi/10.1525/elementa.302/112843/Tropospheric-Ozone-Assessment-Report-Present-day.

Mitchell, M., and Coauthors, 2019: Model Cards for Model Reporting. *Proceedings of the Conference on Fairness, Accountability, and Transparency*, ACM, Atlanta GA USA, 220–229, https://doi.org/10.1145/3287560.3287596, URL https://dl.acm.org/doi/10.1145/3287560.3287596.

Monks, P. S., and Coauthors, 2015: Tropospheric ozone and its precursors from the urban to the global scale from air quality to short-lived climate forcer. *Atmospheric Chemistry and Physics*, **15 (15)**, 8889–8973, https://doi.org/10.5194/acp-15-8889-2015, URL https://acp.copernicus.org/articles/15/8889/2015/.

Munafò, M. R., and Coauthors, 2017: A manifesto for reproducible science. *Nature Human Behaviour*, **1 (1)**, 0021, https://doi.org/10.1038/s41562-016-0021, URL https://www.nature.com/articles/s41562-016-0021.

Munir, S., H. Chen, and K. Ropkins, 2012: Modelling the impact of road traffic on ground level ozone concentration using a quantile regression approach. *Atmospheric Environment*, **60**, 283–291, https://doi.org/10.1016/j.atmosenv.2012.06.043, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231012006061.

Nagi, J., and Coauthors, 2011: Max-pooling convolutional neural networks for vision-based hand gesture recognition. *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*, IEEE, Kuala Lumpur, Malaysia, 342–347, https://doi.org/10.1109/ICSIPA.2011.6144164, URL http://ieeexplore.ieee.org/document/6144164/.

Nature, 2014: Code share. *Nature*, **514 (7524)**, 536–536, https://doi.org/10.1038/514536a, URL http://www.nature.com/articles/514536a.

Navares, R., and J. L. Aznarte, 2020: Predicting air quality with deep learning LSTM: Towards comprehensive models. *Ecological Informatics*, **55**, 101 019, https://doi.org/10.1016/j.ecoinf.2019.101019, URL https://linkinghub.elsevier.com/retrieve/pii/S1574954119303309.

NISO CRediT Working Group, 2022: ANSI/NISO Z39.104-2022, CRediT, Contributor Roles Taxonomy. Tech. rep., National Information Standards Organization, Baltimore, Maryland, U.S.A. https://doi.org/10.3789/ansi.niso.z39.104-2022, URL http://www.niso.org/publications/z39104-2022-credit.

Oppenheim, A. V., and R. W. Schafer, 1975: *Digital signal processing*. Publication Title: Research supported by the Massachusetts Institute of Technology, Bell Telephone Laboratories, and Guggenheim Foundation. Englewood Cliffs, N. J., Prentice-Hall, Inc., 1975. 598 p.

Otero, N., J. Sillmann, J. L. Schnell, H. W. Rust, and T. Butler, 2016: Synoptic and meteorological drivers of extreme ozone concentrations over Europe. *Environmental Research Letters*, **11 (2)**, 024 005, https://doi.org/10.1088/1748-9326/11/2/024005, URL https://iopscience.iop.org/article/10.1088/1748-9326/11/2/024005.

Otero, N., and Coauthors, 2018: A multi-model comparison of meteorological drivers of surface ozone over Europe. *Atmospheric Chemistry and Physics*, **18 (16)**, 12 269–12 288, https://doi.org/10.5194/acp-18-12269-2018, URL https://acp.copernicus.org/articles/18/12269/2018/.

Pascanu, R., T. Mikolov, and Y. Bengio, 2013: On the difficulty of training recurrent neural networks. *Proceedings of the 30th International Conference on Machine Learning*, S. Dasgupta, and D. McAllester, Eds., PMLR, Atlanta, Georgia, USA, Proceedings of Machine Learning Research, Vol. 28, 1310–1318, URL https://proceedings.mlr.press/v28/pascanu13.html, issue: 3.

Pathak, J., and Coauthors, 2022: FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators. *arXiv*, URL http://arxiv.org/abs/2202.11214.

Pearce, J. L., J. Beringer, N. Nicholls, R. J. Hyndman, P. Uotila, and N. J. Tapper, 2011: Investigating the influence of synoptic-scale meteorology on air quality using self-organizing maps and generalized additive modelling. *Atmospheric Environment*, **45 (1)**, 128–136, https://doi.org/10.1016/j.atmosenv.2010.09.032, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231010008071.

Pineau, J., P. Vincent-Lamarre, K. Sinha, V. Lariviere, A. Beygelzimer, F. d'Alche Buc, E. Fox, and H. Larochelle, 2021: Improving Reproducibility in Machine Learning Research(A Report from the NeurIPS 2019 Reproducibility Program). *Journal of Machine Learning Research*, **22 (164)**, 1–20, URL http://jmlr.org/papers/v22/20-303.html.

Porter, W. C., S. A. Safieddine, and C. L. Heald, 2017: Impact of aromatics and monoterpenes on simulated tropospheric ozone and total OH reactivity. *Atmospheric Environment*, **169**, 250–257, https://doi.org/10.1016/j.atmosenv.2017.08.048, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231017305617.

Prybutok, V. R., J. Yi, and D. Mitchell, 2000: Comparison of neural network models with ARIMA and regression models for prediction of Houston's daily maximum ozone concentrations. *European Journal of Operational Research*, **122 (1)**, 31–40, https://doi.org/10.1016/S0377-2217(99)00069-7, URL https://linkinghub.elsevier.com/retrieve/pii/S0377221799000697.

Raina, R., A. Madhavan, and A. Y. Ng, 2009: Large-scale deep unsupervised learning using graphics processors. *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, ACM Press, Montreal, Quebec, Canada, 1–8, https://doi.org/10.1145/1553374.1553486, URL http://portal.acm.org/citation.cfm?doid=1553374.1553486.

Rao, S., P. Porter, J. Mobley, and F. Hurley, 2011: Understanding the spatio-temporal: Variability in air pollution concentrations. *EM: Air and Waste Management Association's Magazine for Environmental Managers*, 42–48.

Rao, S. T., H. Luo, M. Astitha, C. Hogrefe, V. Garcia, and R. Mathur, 2020: On the limit to the accuracy of regional-scale air quality models. *Atmospheric Chemistry and Physics*, **20 (3)**, 1627–1639, https://doi.org/10.5194/acp-20-1627-2020, URL https://acp.copernicus.org/articles/20/1627/2020/.

Rao, S. T., and I. G. Zurbenko, 1994: Detecting and Tracking Changes in Ozone Air Quality. *Air & Waste*, **44 (9)**, 1089–1092, https://doi.org/10.1080/10473289.1994.10467303, URL http://www.tandfonline.com/doi/abs/10.1080/10473289.1994.10467303.

Rao, S. T., I. G. Zurbenko, R. Neagu, P. S. Porter, J. Y. Ku, and R. F. Henry, 1997: Space and Time Scales in Ambient Ozone Data. *Bulletin of the American Meteorological Society*, **78 (10)**, 2153–2166, https://doi.org/10.1175/1520-0477(1997)078<2153:SATSIA>2.0.CO;2, URL http://journals.ametsoc.org/doi/10.1175/1520-0477(1997)078<2153:SATSIA>2.0.CO;2.

Rasp, S., P. D. Dueben, S. Scher, J. A. Weyn, S. Mouatadid, and N. Thuerey, 2020: WeatherBench: A Benchmark Data Set for Data-Driven Weather Forecasting. *Journal of Advances in Modeling Earth Systems*, **12 (11)**, https://doi.org/10.1029/2020MS002203, URL https://onlinelibrary.wiley.com/doi/10.1029/2020MS002203.

Reichstein, M., G. Camps-Valls, B. Stevens, M. Jung, J. Denzler, N. Carvalhais, and Prabhat, 2019: Deep learning and process understanding for data-driven Earth system science. *Nature*, **566 (7743)**, 195–204, https://doi.org/10.1038/s41586-019-0912-1, URL http://www.nature.com/articles/s41586-019-0912-1.

Research, N., 2022: Reporting standards and availability of data, materials, code and protocols. Nature, URL https://www.nature.com/nature-portfolio/editorial-policies/reporting-standards.

Romieu, I., and Coauthors, 2012: Multicity study of air pollution and mortality in Latin America (the ESCALA study). *Research Report (Health Effects Institute)*, **(171)**, 5–86, URL https://pubmed.ncbi.nlm.nih.gov/23311234/.

Ronneberger, O., P. Fischer, and T. Brox, 2015: U-Net: Convolutional Networks for Biomedical Image Segmentation. *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, Eds., Vol. 9351, Springer International Publishing, Cham, 234–241, https://doi.org/10.1007/978-3-319-24574-4_28, URL http://link.springer.com/10.1007/978-3-319-24574-4_28, series Title: Lecture Notes in Computer Science.

Russell, A. R., L. C. Valin, and R. C. Cohen, 2012: Trends in OMI NO2 observations over the United States: effects of emission control technology and the economic recession. *Atmospheric Chemistry and Physics*, **12 (24)**, 12 197–12 209, https://doi.org/10.5194/acp-12-12197-2012, URL https://acp.copernicus.org/articles/12/12197/2012/.

Ryan, W., 1995: Forecasting severe ozone episodes in the Baltimore metropolitan area. *Atmospheric Environment*, **29 (17)**, 2387–2398, https://doi.org/10.1016/1352-2310(94)00302-2, URL https://linkinghub.elsevier.com/retrieve/pii/1352231094003022.

Sayeed, A., Y. Choi, E. Eslami, Y. Lops, A. Roy, and J. Jung, 2020: Using a deep convolutional neural network to predict 2017 ozone concentrations, 24 hours in advance. *Neural Networks*, **121**, 396–408, https://doi.org/10.1016/j.neunet.2019.09.033, URL https://linkinghub.elsevier.com/retrieve/pii/S0893608019303156.

Sayeed, A., Y. Choi, A. Pouyaei, Y. Lops, J. Jung, and A. K. Salman, 2022: CNN-based model for the spatial imputation (CMSI version 1.0) of in-situ ozone and PM2.5 measurements. *Atmospheric Environment*, **289**, 119 348, https://doi.org/10.1016/j.atmosenv.2022.119348, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231022004137.

Schaap, M., and Coauthors, 2015: Performance of European chemistry transport models as function of horizontal resolution. *Atmospheric Environment*, **112**, 90–105, https://doi.org/10.1016/j.atmosenv.2015.04.003, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231015300066.

Scherer, D., A. Müller, and S. Behnke, 2010: Evaluation of Pooling Operations in Convolutional Architectures for Object Recognition. *Artificial Neural Networks – ICANN 2010*, K. Diamantaras, W. Duch, and L. S. Iliadis, Eds., Springer Berlin Heidelberg, Berlin, Heidelberg, 92–101, https://doi.org/10.1007/978-3-642-15825-4_10.

Schlink, U., O. Herbarth, M. Richter, S. Dorling, G. Nunnari, G. Cawley, and E. Pelikan, 2006: Statistical models to assess the health effects and to forecast ground-level ozone. *Environmental Modelling & Software*, **21 (4)**, 547–558, https://doi.org/10.1016/j.envsoft.2004.12.002, URL https://linkinghub.elsevier.com/retrieve/pii/S1364815204003196.

Schmidt, R. M., F. Schneider, and P. Hennig, 2021: Descending through a Crowded Valley - Benchmarking Deep Learning Optimizers. *Proceedings of the 38th International Conference on Machine Learning*, M. Meila, and T. Zhang, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 139, 9367–9376, URL https://proceedings.mlr.press/v139/schmidt21a.html.

Schultz, M. G., C. Betancourt, B. Gong, F. Kleinert, M. Langguth, L. H. Leufen, A. Mozaffari, and S. Stadtler, 2021: Can deep learning beat numerical weather prediction? *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, **379 (2194)**, 20200 097, https://doi.org/10.1098/rsta.2020.0097, URL https://royalsocietypublishing.org/doi/10.1098/rsta.2020.0097.

Schultz, M. G., and Coauthors, 2017: Tropospheric Ozone Assessment Report: Database and metrics data of global surface ozone observations. *Elementa: Science of the Anthropocene*, **5**, 58, https://doi.org/10.1525/elementa.244, URL https://online.ucpress.edu/elementa/article/doi/10.1525/elementa.244/112447/Tropospheric-Ozone-Assessment-Report-Database-and.

Seinfeld, J. H., and S. N. Pandis, 2016: *Atmospheric chemistry and physics: from air pollution to climate change*. Third edition ed., John Wiley & Sons, Inc, Hoboken, New Jersey.

Seltzer, K. M., D. T. Shindell, G. Faluvegi, and L. T. Murray, 2017: Evaluating Modeled Impact Metrics for Human Health, Agriculture Growth, and Near-Term Climate: EVALUATING MODELED IMPACT METRICS. *Journal of Geophysical Research: Atmospheres*, **122 (24)**, 13,506–13,524, https://doi.org/10.1002/2017JD026780, URL http://doi.wiley.com/10.1002/2017JD026780.

Seltzer, K. M., D. T. Shindell, P. Kasibhatla, and C. S. Malley, 2020: Magnitude, trends, and impacts of ambient long-term ozone exposure in the United States from 2000 to 2015. *Atmospheric Chemistry and Physics*, **20 (3)**, 1757–1775, https://doi.org/10.5194/acp-20-1757-2020, URL https://acp.copernicus.org/articles/20/1757/2020/.

Serra-Garcia, M., and U. Gneezy, 2021: Nonreplicable publications are cited more than replicable ones. *Science Advances*, **7 (21)**, eabd1705, https://doi.org/10.1126/sciadv.abd1705, URL https://www.science.org/doi/10.1126/sciadv.abd1705.

Shindell, D., G. Faluvegi, P. Kasibhatla, and R. Van Dingenen, 2019: Spatial Patterns of Crop Yield Change by Emitted Pollutant. *Earth's Future*, **7 (2)**, 101–112, https://doi.org/10.1029/2018EF001030, URL https://onlinelibrary.wiley.com/doi/10.1029/2018EF001030.

Sivaprasad, P. T., F. Mai, T. Vogels, M. Jaggi, and F. Fleuret, 2020: Optimizer Benchmarking Needs to Account for Hyperparameter Tuning. *Proceedings of the 37th International Conference on Machine Learning*, JMLR.org, ICML'20.

Siwek, K., and S. Osowski, 2016: Data mining methods for prediction of air pollution. *International Journal of Applied Mathematics and Computer Science*, **26 (2)**, 467–478, https://doi.org/10.1515/amcs-2016-0033, URL https://www.sciendo.com/article/10.1515/amcs-2016-0033.

Solazzo, E., and Coauthors, 2017: Evaluation and error apportionment of an ensemble of atmospheric chemistry transport modeling systems: multivariable temporal and spatial breakdown. *Atmospheric Chemistry and Physics*, **17 (4)**, 3001–3054, https://doi.org/10.5194/acp-17-3001-2017, URL https://acp.copernicus.org/articles/17/3001/2017/.

Solberg, S., A. Colette, and C. Guerreiro, 2016: Discounting the impact of meteorology to the ozone concentration trends. *European Topic Centre on Air Pollution and Climate Change Mitigation, Bilthoven, the Netherlands, Technical Paper*, **9 (2015/09)**.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, 2014: Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, **15 (56)**, 1929–1958, URL http://jmlr.org/papers/v15/srivastava14a.html.

Srivastava, R. K., K. Greff, and J. Schmidhuber, 2015: Training Very Deep Networks. *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds., Curran Associates, Inc., Vol. 28, URL https://proceedings.neurips.cc/paper/2015/file/215a71a12769b056c3c32e7299f1c5ed-Paper.pdf.

Stock, Z. S., M. R. Russo, and J. A. Pyle, 2014: Representing ozone extremes in European megacities: the importance of resolution in a global chemistry climate model. *Atmospheric Chemistry and Physics*, **14 (8)**, 3899–3912, https://doi.org/10.5194/acp-14-3899-2014, URL https://acp.copernicus.org/articles/14/3899/2014/.

Szegedy, C., and Coauthors, 2015: Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, Boston, MA, USA, 1–9, https://doi.org/10.1109/CVPR.2015.7298594, URL http://ieeexplore.ieee.org/document/7298594/.

Tatman, R., J. VanderPlas, and S. Dane, 2018: A practical taxonomy of reproducibility for machine learning research. *The 2nd Reproducibility in Machine Learning Workshop at ICML 2018*, Stockholm, Sweden, URL https://openreview.net/pdf?id=B1eYYK5QgX.

US EPA, 2020: Integrated Science Assessment (ISA) for Ozone and Related Photochemical Oxidants (Final Report, April 2020). US Environmental Protection Agency Washington, DC, USA.

van den Oord, A., and Coauthors, 2016: WaveNet: A Generative Model for Raw Audio. *arXiv*, https://doi.org/10.48550/ARXIV.1609.03499, URL https://arxiv.org/abs/1609.03499.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, 2017: Attention is All you Need. *Advances in Neural Information Processing Systems*,

I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., Vol. 30, URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Vautard, R., and Coauthors, 2012: Evaluation of the meteorological forcing used for the Air Quality Model Evaluation International Initiative (AQMEII) air quality simulations. *Atmospheric Environment*, **53**, 15–37, https://doi.org/10.1016/j.atmosenv.2011.10.065, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231011011605.

Wang, H.-W., X.-B. Li, D. Wang, J. Zhao, H.-d. He, and Z.-R. Peng, 2020: Regional prediction of ground-level ozone using a hybrid sequence-to-sequence deep learning approach. *Journal of Cleaner Production*, **253**, 119 841, https://doi.org/10.1016/j.jclepro.2019.119841, URL https://linkinghub.elsevier.com/retrieve/pii/S0959652619347110.

Wang, S. W., H. Levy, G. Li, and H. Rabitz, 1999: Fully equivalent operational models for atmospheric chemical kinetics within global chemistry-transport models. *Journal of Geophysical Research: Atmospheres*, **104 (D23)**, 30 417–30 426, https://doi.org/10.1029/1999JD900830, URL http://doi.wiley.com/10.1029/1999JD900830.

Watson-Parris, D., and Coauthors, 2022: ClimateBench v1.0: A Benchmark for Data-Driven Climate Projections. *Journal of Advances in Modeling Earth Systems*, **14 (10)**, https://doi.org/10.1029/2021MS002954, URL https://onlinelibrary.wiley.com/doi/10.1029/2021MS002954.

Weichselbaum, F., 2022: Deep neural network techniques for weather forecasting: An implementation of a short-term forecast on different locations. Master's thesis, Rhenish Friedrich Wilhelm University of Bonn, Bonn, Germany.

Wells, B., P. Dolwick, B. Eder, M. Evangelista, K. Foley, E. Mannshardt, C. Misenis, and A. Weishampel, 2021: Improved estimation of trends in U.S. ozone concentrations adjusted for interannual variability in meteorological conditions. *Atmospheric Environment*, **248**, 118 234, https://doi.org/10.1016/j.atmosenv.2021.118234, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231021000522.

WHO, 2013: *Review of evidence on health aspects of air pollution: REVIHAAP project: technical report*. World Health Organization. Regional Office for Europe, URL https://apps.who.int/iris/handle/10665/345329.

WHO, 2021: *WHO global air quality guidelines: particulate matter (PM2.5 and PM10), ozone, nitrogen dioxide, sulfur dioxide and carbon monoxide*. World Health Organization, URL https://apps.who.int/iris/handle/10665/345329.

Wilkinson, M. D., and Coauthors, 2016: The FAIR Guiding Principles for scientific data management and stewardship. *Scientific Data*, **3 (1)**, 160 018, https://doi.org/10.1038/sdata.2016.18, URL http://www.nature.com/articles/sdata201618.

Wilks, D. S., 2006: *Statistical Methods in the Atmospheric Sciences*. 2nd ed., International geophysics series, Elsevier Academic Press, Amsterdam.

Wilson, A. C., R. Roelofs, M. Stern, N. Srebro, and B. Recht, 2017: The Marginal Value of Adaptive Gradient Methods in Machine Learning. *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., Curran Associates, Inc., Vol. 30, URL https://proceedings.neurips.cc/paper/2017/file/81b3833e2504647f9d794f7d7b9bf341-Paper.pdf.

Wise, E. K., and A. C. Comrie, 2005: Extending the Kolmogorov–Zurbenko Filter: Application to Ozone, Particulate Matter, and Meteorological Trends. *Journal of the Air & Waste Management Association*, **55 (8)**, 1208–1216, https://doi.org/10.1080/10473289.2005.10464718, URL https://www.tandfonline.com/doi/full/10.1080/10473289.2005.10464718.

Yang, W., and I. Zurbenko, 2010: Kolmogorov-Zurbenko filters: Kolmogorov-Zurbenko Filters. *Wiley Interdisciplinary Reviews: Computational Statistics*, **2 (3)**, 340–351, https://doi.org/10.1002/wics.71, URL https://onlinelibrary.wiley.com/doi/10.1002/wics.71.

Young, P. J., and Coauthors, 2018: Tropospheric Ozone Assessment Report: Assessment of global-scale model performance for global and regional ozone distributions, variability, and trends. *Elementa: Science of the Anthropocene*, **6**, 10, https://doi.org/10.1525/elementa.265, URL https://online.ucpress.edu/elementa/article/doi/10.1525/elementa.265/112813/Tropospheric-Ozone-Assessment-Report-Assessment-of.

Yu, F., and V. Koltun, 2016: Multi-Scale Context Aggregation by Dilated Convolutions. *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio, and Y. LeCun, Eds., URL http://arxiv.org/abs/1511.07122.

Zhan, Y., Y. Luo, X. Deng, M. L. Grieneisen, M. Zhang, and B. Di, 2018: Spatiotemporal prediction of daily ambient ozone levels across China using random forest for human exposure assessment. *Environmental Pollution*, **233**, 464–473, https://doi.org/10.1016/j.envpol.2017.10.029, URL https://linkinghub.elsevier.com/retrieve/pii/S0269749117328907.

Zhang, Y., M. Bocquet, V. Mallet, C. Seigneur, and A. Baklanov, 2012: Real-time air quality forecasting, part I: History, techniques, and current status. *Atmospheric Environment*, **60**, 632–655, https://doi.org/10.1016/j.atmosenv.2012.06.031, URL https://linkinghub.elsevier.com/retrieve/pii/S1352231012005900.

Zhang, Y., O. R. Cooper, A. Gaudel, A. M. Thompson, P. Nédélec, S.-Y. Ogino, and J. J. West, 2016: Tropospheric ozone change from 1980 to 2010 dominated by equatorward redistribution of emissions. *Nature Geoscience*, **9 (12)**, 875–879, https://doi.org/10.1038/ngeo2827, URL http://www.nature.com/articles/ngeo2827.

Zhang, Y., and Coauthors, 2018: Long-term trends in the ambient PM2.5- and O3-related mortality burdens in the United States under emission reductions from 1990 to 2010. *Atmospheric Chemistry and Physics*, **18 (20)**, 15 003–15 016, https://doi.org/10.5194/acp-18-15003-2018, URL https://acp.copernicus.org/articles/18/15003/2018/.

Zhao, J., F. Huang, J. Lv, Y. Duan, Z. Qin, G. Li, and G. Tian, 2020: Do RNN and LSTM have Long Memory? *Proceedings of the 37th International Conference on Machine Learning*, H. D. III, and A. Singh, Eds., PMLR, Proceedings of Machine Learning Research, Vol. 119, 11 365–11 375, URL https://proceedings.mlr.press/v119/zhao20c.html.

Zhou, Z., M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, 2018: UNet++: A Nested U-Net Architecture for Medical Image Segmentation. *arXiv*, URL http://arxiv.org/abs/1807.10165.

Zurbenko, I. G., 1986: *The spectral analysis of time series*. North-Holland series in statistics and probability; 2, North-Holland, Amsterdam, URL http://www.gbv.de/dms/hbz/toc/ht002861599.pdf.

B. Acronyms

AAAI	Association for the Advancement of AI.
AI	artificial intelligence.
AQS	air quality station.
CAMS	Copernicus Atmosphere Monitoring Service.
CNN	convolutional neural network.
CPU	central processing unit.
CRediT	Contributor Roles Taxonomy.
CTM	chemical transport model.
DL	deep learning.
dma8	daily maximum 8-hour running average.
DU	diurnal.
ECMWF	European Centre for Medium-Range Weather Forecasts.
ERA5	ECMMF Reanalysis of the fifth Generation.
EU	European Union.
FIR	finite impulse response.
FNN	feedforward neural network.
GAN	generative adversarial network.
GMD	Geoscientific Model Development.
GPU	graphical processing unit.
GRU	gated recurrent unit.
HPC	high performance computing.
ID	intraday.
IJCAI	International Joint Conference on AI.
KZF	Kolmogorov-Zurbenko filter.
LSTM	long short-term memory.
LT	long-term.
ME	mean error.
ML	machine learning.
MLAir	Machine Learning on Air data.
MSE	mean squared error.
NAQPMS	Nested Air Quality Prediction Modeling System.
NIPS	Neural Information Processing Systems.
NMVOC	non-methane volatile organic compound.
NN	neural network.
REA6	high-resolution reanalysis system COSMO-REA6.
ReLU	rectified linear unit.
ResNet	residual neural network.
RMSE	root mean squared error.
RNN	recurrent neural network.
SE	seasonal.
SGD	stochastic gradient descent.
ST	short-term.
SY	synoptic.
tanh	hyperbolic tangent.
TFT	temporal fusion transformer.
TOAR DB	Tropospheric Ozone Assessment Report database.
VAE	variational autoencoder.
VOC	volatile organic compound.
WHO	World Health Organization.

C. Acknowledgements

This work heralds the end of just over four exciting years. Along the way, I have enjoyed the support of colleagues, family and acquaintances in a plethora of ways. Because they played an important role throughout the course of my work, I would like to take this opportunity to express my heartfelt gratitude for their support. My special thanks go to the following people:

- PD Dr. Martin G. Schultz for supervising my PhD project, providing support, innumerable ideas, teaching me writing and storytelling skills, offering to co-author the PhilTrans paper and enabling me to participate in several scientific conferences in Austria, France, UK and US,
- Prof. Dr. Andreas Hense for his support and co-supervision of my PhD,
- Felix Kleinert for the great and collegial work together, the close collaboration on the MLAir repository, the daily meetings at the *Heißer Draht Bonn Stuttgart* in the morning with almost 600 meetings over the last years, and for the continuous hospitality when I visited Bonn,
- Marco Bernreuther for many detailed discussions on ML and the tremendous review of this thesis with numerous helpful and mathematically profound comments,
- Birgitta Weltermann and Martin Erdmann for being my personal advisory board, sharing interesting ideas and methods and always giving me motivational support,
- Dr. Michael Langguth for the detailed review of key parts in this thesis and, of course, also for the meetings in the *Heißer Draht*,
- Hubert and Maria Leufen for accommodation close to the workplace and a lovely and so stirring time in the multigenerational house,
- Dr. Marleen Braun and Dr. Amir Mozaffari for reviewing selected parts of the thesis,
- the ESDE group in Jülich for the time spent together and especially Sabine Schröder for always having a sympathetic ear for any problems and, in particular, for questions regarding TOAR DB,
- my former colleague Jessica Ahring for all her joy and motivation, although much of the technical work was often very exhausting,
- and last but not least very especially to my parents, Christiane and Heinrich, as well as my family, Céline, Bruno and my family-in-law for their support, distraction, motivation, discussions and anything else they did.

D. Manuscripts

D.1. Leufen et al. (2021): *MLAir (v1.0) – a tool to enable fast and flexible machine learning on air data time series*

Geosci. Model Dev., 14, 1553–1574, 2021
https://doi.org/10.5194/gmd-14-1553-2021

Geoscientific
Model Development

MLAir (v1.0) – a tool to enable fast and flexible machine learning on air data time series

Lukas Hubert Leufen[1,2], **Felix Kleinert**[1,2], **and Martin G. Schultz**[1]

[1]Jülich Supercomputing Centre, Research Centre Jülich, Jülich, Germany
[2]Institute of Geosciences, Rhenish Friedrich Wilhelm University of Bonn, Bonn, Germany

Correspondence: Lukas Hubert Leufen (l.leufen@fz-juelich.de)

Received: 9 October 2020 – Discussion started: 23 October 2020
Revised: 29 January 2021 – Accepted: 2 February 2021 – Published: 17 March 2021

Abstract. With MLAir (Machine Learning on Air data) we created a software environment that simplifies and accelerates the exploration of new machine learning (ML) models, specifically shallow and deep neural networks, for the analysis and forecasting of meteorological and air quality time series. Thereby MLAir is not developed as an abstract workflow, but hand in hand with actual scientific questions. It thus addresses scientists with either a meteorological or an ML background. Due to their relative ease of use and spectacular results in other application areas, neural networks and other ML methods are also gaining enormous momentum in the weather and air quality research communities. Even though there are already many books and tutorials describing how to conduct an ML experiment, there are many stumbling blocks for a newcomer. In contrast, people familiar with ML concepts and technology often have difficulties understanding the nature of atmospheric data. With MLAir we have addressed a number of these pitfalls so that it becomes easier for scientists of both domains to rapidly start off their ML application. MLAir has been developed in such a way that it is easy to use and is designed from the very beginning as a stand-alone, fully functional experiment. Due to its flexible, modular code base, code modifications are easy and personal experiment schedules can be quickly derived. The package also includes a set of validation tools to facilitate the evaluation of ML results using standard meteorological statistics. MLAir can easily be ported onto different computing environments from desktop workstations to high-end supercomputers with or without graphics processing units (GPUs).

1 Introduction

In times of rising awareness of air quality and climate issues, the investigation of air quality and weather phenomena is moving into focus. Trace substances such as ozone, nitrogen oxides, or particulate matter pose a serious health hazard to humans, animals, and nature (Cohen et al., 2005; Bentayeb et al., 2015; World Health Organization, 2013; Lefohn et al., 2018; Mills et al., 2018; US Environmental Protection Agency, 2020). Accordingly, the analysis and prediction of air quality are of great importance in order to be able to initiate appropriate countermeasures or issue warnings. The prediction of weather and air quality has been established operationally in many countries and has become a multi-million dollar industry, creating and selling specialized data products for many different target groups.

These days, forecasts of weather and air quality are generally made with the help of so-called Eulerian grid point models. This type of model, which solves physical and chemical equations, operates on grid structures. In fact, however, local observations of weather and air quality are strongly influenced by the immediate environment. Such local influences are quite difficult for atmospheric chemistry models to accurately simulate due to the limited grid resolution of these models and because of uncertainties in model parameterizations. Consequently, both global models and so-called small-scale models, whose grid resolution is still in the magnitude of about a kilometre and thus rather coarse in comparison to local-scale phenomena in the vicinity of a measurement site, show a high uncertainty of the results (see Vautard, 2012; Brunner et al., 2015). To enhance the model output, approaches focusing on the individual point measurements

Published by Copernicus Publications on behalf of the European Geosciences Union.

at weather and air quality monitoring stations through downscaling methods are applied allowing local effects to be taken into account. Unfortunately, these methods, being optimized for specific locations, cannot be generalized for other regions and need to be re-trained for each measurement site.

Recently, a variety of machine learning (ML) methods have been developed to complement the traditional downscaling techniques. Such methods (e.g. neural networks, random forest) are able to recognize and reproduce underlying and complex relationships in data sets. Driven in particular by computer vision and speech recognition, technologies like convolutional neural networks (CNNs; Lecun et al., 1998), or recurrent networks variations such as long short-term memory (LSTM; Hochreiter and Schmidhuber, 1997) or gated recurrent units (GRUs; Cho et al., 2014) but also more advanced concepts like variational autoencoders (VAEs; Kingma and Welling, 2014; Rezende et al., 2014), or generative adversarial networks (GANs; Goodfellow et al., 2014), are powerful and widely and successfully used. The application of such methods to weather and air quality data is rapidly gaining momentum.

Although the scientific areas of ML and atmospheric science have existed for many years, combining both disciplines is still a formidable challenge, because scientists from these areas do not speak the same language. Atmospheric scientists are used to building models on the basis of physical equations and empirical relationships from field experiments, and they evaluate their models with data. In contrast, data scientists use data to build their models on and evaluate them either with additional independent data or physical constraints. This elementary difference can lead to misinterpretation of study results so that, for example, the ability of the network to generalize is misjudged. Another problem of several published studies on ML approaches to weather forecasting is an incomplete reporting of ML parameters, hyperparameters, and data preparation steps that are key to comprehending and reproducing the work that was done. As shown by Musgrave et al. (2020) these issues are not limited to meteorological applications of ML only.

To further advance the application of ML in atmospheric science, easily accessible solutions to run and document ML experiments together with readily available and fully documented benchmark data sets are urgently needed (see Schultz et al., 2021). Such solutions need to be understandable by both communities and help both sides to prevent unconscious blunders. A well-designed workflow embedded in a meteorological and ML-related environment while accomplishing subject-specific requirements will bring forward the usage of ML in this specific research area.

In this paper, we present a new framework to enable fast and flexible Machine Learning on Air data time series (MLAir). Fast means that MLAir is distributed as full end-to-end framework and thereby simple to deploy. It also allows typical optimization techniques to be deployed in ML workflows and offers further technical features like the use of graphics processing units (GPUs) due to the underlying ML library. MLAir is suitable for ML beginners due to its simple usage but also offers high customization potential for advanced ML users. It can therefore be employed in real-world applications. For example, more complex model architectures can be easily integrated. ML experts who want to explore weather or air quality data will find MLAir helpful as it enforces certain standards of the meteorological community. For example, its data preparation step acknowledges the autocorrelation which is typically seen in meteorological time series, and its validation package reports well-established skill scores, i.e. improvement of the forecast compared to reference models such as persistence and climatology. From a software design perspective, MLAir has been developed according to state-of-the-art software development practices.

This article is structured as follows. Section 2 introduces MLAir by expounding the general design behind the MLAir workflow. We also share a few more general points about ML and what a typical workflow looks like. This is followed by Sect. 3 showing three application examples to allow the reader to get a general understanding of the tool. Furthermore, we show how the results of an experiment conducted by MLAir are structured and which statistical analysis is applied. Section 4 extends further into the configuration options of an experiment and details on customization. Section 5 delineates the limitations of MLAir and discusses for which applications the tool might not be suitable. Finally, Sect. 6 concludes with an overview and outlook on planned developments for the future.

At this point we would like to point out that in order to simplify the readability of the paper, highlighting is used. *Frameworks* are highlighted in italics and typewriter font is used for `code` elements such as class names or variables. Other expressions that, for example, describe a class but do not explicitly name it, are not highlighted at all in the text. Last but not least, we would like to mention that *MLAir* is an open-source project and contributions from all communities are welcome.

2 MLAir workflow and design

ML in general is the application of a learning algorithm to a data set whereby a statistical model describing relations within the data is generated. During the so-called training process, the model learns patterns in the data set with the aid of the learning algorithm. Afterwards, this model can be applied to new data. Since there is a large number of learning algorithms and also an arbitrarily large number of different ML architectures, it is generally not possible to determine in advance which approach will deliver the best results under which configuration. Therefore, the optimal setting must be found by trial and error.

ML experiments usually follow similar patterns. First, data must be obtained, cleaned if necessary, and finally put into a suitable format (preprocessing). Next, an ML model is selected and configured (model setup). Then the learning algorithm can optimize the model under the selected settings on the data. This optimization is an iterative procedure and each iteration is called an epoch (training). The accuracy of the model is then evaluated (validation). If the results are not satisfactory, the experiment is continued with modified settings (i.e. hyperparameters) or started again with a new model. For further details on ML, we refer to Bishop (2006) and Goodfellow et al. (2016) but would also like to point out that there is a large amount of further introductory literature and freely available blog entries and videos, and that the books mentioned here are only two of many options out there.

The overall goal of designing *MLAir* was to create a ready-to-run ML application for the task of forecasting weather and air quality time series. The tool should allow many customization options to enable users to easily create a custom ML workflow, while at the same time it should support users in executing ML experiments properly and evaluate their results according to accepted standards of the meteorological community. At this point, it is pertinent to recall that *MLAir*'s current focus is on neural networks.

In this section we present the general concepts on which *MLAir* is based. We first comment on the choice of the underlying programming language and the packages and frameworks used (Sect. 2.1). We then focus on the design considerations and choices and introduce the general workflow of *MLAir* (Sect. 2.2). Thereafter we explain how the concepts of run modules (Sect. 2.3), model class (Sect. 2.4), and data handler (Sect. 2.5) were conceived and how these modules interact with each other. More detailed information on, for example, how to adapt these modules can be found in the corresponding subsection of the later Sect. 4.

2.1 Coding language

Python (Python Software Foundation, 2018, release 3.6.8) was used as the underlying coding language for several reasons. *Python* is pretty much independent of the operating system and code does not need to be compiled before a run. *Python* is flexible to handle different tasks like data loading from the web, training of the ML model or plotting. Numerical operations can be executed quite efficiently due to the fact that they are usually performed by highly optimized and compiled mathematical libraries. Furthermore, because of its popularity in science and economics, *Python* has a huge variety of freely available packages to use. Furthermore, *Python* is currently the language of choice in the ML community (Elliott, 2019) and has well-developed easy-to-use frameworks like *TensorFlow* (Abadi et al., 2015) or *PyTorch* (Paszke et al., 2019) which are state-of-the-art tools to work on ML problems. Due to the presence of such compiled frameworks, there is for instance no performance loss during the training, which is the biggest part of the ML workflow, by using *Python*.

Concerning the ML framework, *Keras* (Chollet et al., 2015, release 2.2.4) was chosen for the ML parts using *TensorFlow* (release 1.13.1) as back-end. *Keras* is a framework that abstracts functionality out of its back-end by providing a simpler syntax and implementation. For advanced model architectures and features it is still possible to implement parts or even the entire model in native *TensorFlow* and use the *Keras* front-end for training. Furthermore, *TensorFlow* has GPU support for training acceleration if a GPU device is available on the running system.

For data handling, we chose a combination of *xarray* (Hoyer and Hamman, 2017; Hoyer et al., 2020, release 0.15.0) and *pandas* (Wes McKinney, 2010; Reback et al., 2020, release 1.0.1). *pandas* is an open-source tool to analyse and manipulate data primarily designed for tabular data. *xarray* was inspired by *pandas* and has been developed to work with multi-dimensional arrays as simply and efficiently as possible. *xarray* is based on the off-the-shelf *Python* package for scientific computing *NumPy* (van der Walt et al., 2011, release 1.18.1) and introduces labels in the form of dimensions, coordinates, and attributes on top of raw *NumPy*-like arrays.

2.2 Design of the MLAir workflow

According to the goals outlined above, *MLAir* was designed as an end-to-end workflow comprising all required steps of the time series forecasting task. The workflow of *MLAir* is controlled by a run environment, which provides a central data store, performs logging, and ensures the orderly execution of a sequence of individual stages. Different workflows can be defined and executed under the umbrella of this environment. The standard *MLAir* workflow (described in Sect. 2.3) contains a sequence of typical steps for ML experiments (Fig. 1), i.e. experiment setup, preprocessing, model setup, training, and postprocessing.

Besides the run environment, the experiment setup plays a very important role. During experiment setup, all customization and configuration modules, like the model class (Sect. 2.4), data handler (Sect. 2.5), or hyperparameters, are collected and made available to *MLAir*. Later, during execution of the workflow, these modules are then queried. For example, the hyperparameters are used in training whereas the data handler is already used in the preprocessing. We want to mention that apart from this default workflow, it is also possible to define completely new stages and integrate them into a custom *MLAir* workflow (see Sect. 4.8).

2.3 Run modules

MLAir models the ML workflow as a sequence of self-contained stages called run modules. Each module handles distinct tasks whose calculations or results are usually required for all subsequent stages. At run time, all run modules can interchange information through a temporary data store.

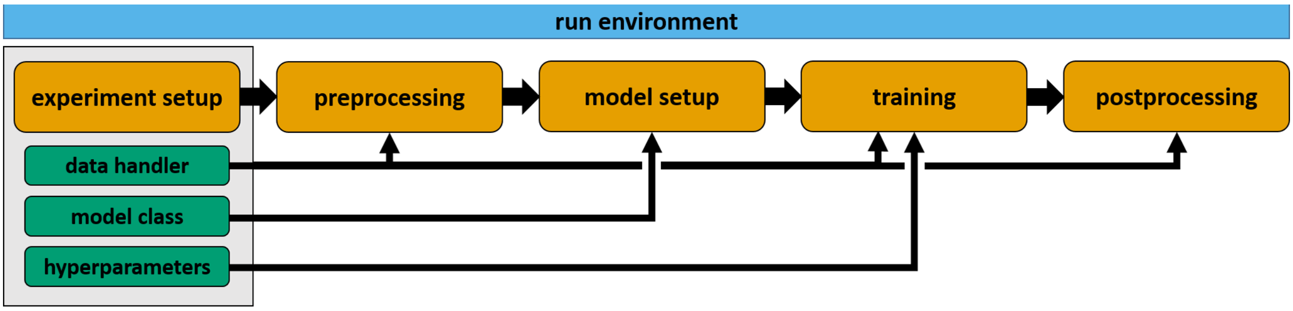


Figure 1. Visualization of the *MLAir* standard setup `DefaultWorkflow` including the stages `ExperimentSetup`, `PreProcessing`, `ModelSetup`, `Training`, and `PostProcessing` (all highlighted in orange) embedded in the `RunEnvironment` (sky blue). Each experiment customization (bluish green) like the data handler, model class, and hyperparameter shown as examples, is set during the initial `ExperimentSetup` and affects various stages of the workflow.

The run modules are executed sequentially in predefined order. A run module is only executed if the previous step was completed without error. More advanced workflow concepts such as conditional execution of run modules are not implemented in this version of *MLAir*. Also, run modules cannot be run in parallel, although a single run module can very well execute parallel code. In the default setup (Fig. 1), the *MLAir* workflow constitutes the following run modules:

– *Run environment.* The run module `RunEnvironment` is the base class for all other run modules. By wrapping the `RunEnvironment` class around all run modules, parameters are tracked, the workflow logging is centralized, and the temporary data store is initialized. After each run module and at the end of the experiment, `RunEnvironment` guarantees a smooth (experiment) closure by providing supplementary information on stage execution and parameter access from the data store.

– *Experiment setup.* The initial stage of *MLAir* to set up the experiment workflow is called `ExperimentSetup`. Parameters which are not customized are filled with default settings and stored for the experiment workflow. Furthermore, all local paths for the experiment and data are created during experiment setup.

– *Preprocessing.* During the run module `PreProcessing`, *MLAir* loads all required data and carries out typical ML preparation steps to have the data ready to use for training. If the `DefaultDataHandler` is used, this step includes downloading or loading of (locally stored) data, data transformation and interpolation. Finally, data are split into the subsets for training, validation, and testing.

– *Model setup.* The `ModelSetup` run module builds the raw ML model implemented as a model class (see Sect. 2.4), sets *Keras* and *TensorFlow* callbacks and checkpoints for the training, and finally compiles the model. Additionally, if using a pre-trained model, the weights of this model are loaded during this stage.

– *Training.* During the course of the `Training` run module, training and validation data are distributed according to the parameter `batch_size` to properly feed the ML model. The actual training starts subsequently. After each epoch of training, the model performance is evaluated on validation data. If performance improves as compared to previous cycles, the model is stored as `best_model`. This `best_model` is then used in the final analysis and evaluation.

– *Postprocessing.* In the final stage, `PostProcessing`, the trained model is statistically evaluated on the test data set. For comparison, *MLAir* provides two additional forecasts, first an ordinary multi-linear least squared fit trained on the same data as the ML model and second a persistence forecast, where observations of the past represent the forecast for the next steps within the prediction horizon. For daily data, the persistence forecast refers to the last observation of each sample to hold for all forecast steps. Skill scores based on the model training and evaluation metric are calculated for all forecasts and compared with climatological statistics. The evaluation results are saved as publication-ready graphics. Furthermore, a bootstrapping technique can be used to evaluate the importance of each input feature. More details on the statistical analysis that is carried out can be found in Sect. 3.3. Finally, a geographical overview map containing all stations is created for convenience.

Ideally this predefined default workflow should meet the requirements for an entire end-to-end ML workflow on station-wise observational data. Nevertheless, *MLAir* provides options to customize the workflow according to the application needs (see Sect. 4.8).

2.4 Model class

In order to ensure a proper functioning of ML models, *MLAir* uses a model class, so that all models are created according to the same scheme. Inheriting from the `AbstractModelClass` guarantees correct handling during the workflow. The model class is designed to follow an easy plug-and-play behaviour so that within this security mechanism, it is possible to create highly customized models with the frameworks *Keras* and *TensorFlow*. We know that wrapping such a class around each ML model is slightly more complicated compared to building models directly in Keras, but by requiring the user to build their models in the style of a model class, the model structure can be documented more easily. Thus, there is less potential for errors when running through an ML workflow, in particular when this is done many times to find out the best model setup, for example. More details on the model class can be found in Sect. 4.5.

2.5 Data handler

In analogy to the model class, the data handler organizes all operations related to data retrieval, preparation and provision of data samples. If a set of observation stations is being examined in the *MLAir* workflow, a new instance of the data handler is created for each station automatically and *MLAir* will take care of the iteration across all stations. As with the creation of a model, it is not necessary to modify *MLAir*'s source code. Instead, every data handler inherits from the `AbstractDataHandler` class which provides guidance on which methods need to be adapted to the actual workflow.

By default, *MLAir* uses the `DefaultDataHandler`. It accesses data from Jülich Open Web Interface (JOIN, Schultz et al., 2017a, b) as demonstrated in Sect. 3.1. A detailed description of how to use this data handler can be found in Sect. 4.4. However, if a different data source or structure is used for an experiment, the `DefaultDataHandler` must be replaced by a custom data handler based on the `AbstractDataHandler`. Simply put, such a custom handler requires methods for creating itself at runtime and methods that return the inputs and outputs. Partitioning according to the batch size or suchlike is then handled by *MLAir* at the appropriate moment and does not need to be integrated into the custom data handler. Further information about custom data handlers follows in Sect. 4.3, and we refer to the source code documentation for additional details.

3 Conducting an experiment with MLAir

Before we dive deeper into available features and the actual implementation, we show three basic examples of the *MLAir* usage to demonstrate the underlying ideas and concepts and how first modifications can be made (Sect. 3.1). In Sect. 3.2, we then explain how the output of an *MLAir* experiment is structured and which graphics are created. Finally, we briefly touch on the statistical part of the model evaluation (Sect. 3.3).

3.1 Running first experiments with MLAir

To install *MLAir*, the program can be downloaded as described in the *Code availability* section, and the *Python* library dependencies should be installed from the requirements file. To test the installation, *MLAir* can be run in a default configuration with no extra arguments (see Fig. 2). These two commands will execute the workflow depicted in Fig. 1. This will perform an ML forecasting experiment of daily maximum ground-level ozone concentrations using a simple feed-forward neural network based on seven input variables consisting of preceding trace gas concentrations of ozone and nitrogen dioxide, and the values of temperature, humidity, wind speed, cloud cover, and the planetary boundary layer height.

MLAir uses the `DefaultDataHandler` class (see Sect. 4.4) if not explicitly stated and automatically starts downloading all required air quality and meteorological data from JOIN the first time it is executed after a fresh installation. This web interface provides access to a database of measurements of over 10 000 air quality monitoring stations worldwide, assembled in the context of the Tropospheric Ozone Assessment Report (TOAR, 2014–2021). In the default configuration, 21-year time series of nine variables from five stations are retrieved with a daily aggregated resolution (see Table 3 for details on aggregation). The retrieved data are stored locally to save time on the next execution (the data extraction can of course be configured as described in Sect. 4.4).

After preprocessing of the data, splitting them into training, validation, and test data, and converting them to a *xarray* and *NumPy* format (details in Sect. 2.1), *MLAir* creates a new vanilla feed-forward neural network and starts to train it. The training is finished after a fixed number of epochs. In the default settings, the `epochs` parameter is preset to 20. Finally, the results are evaluated according to meteorological standards and a default set of plots is created. The trained model, all results and forecasts, the experiment parameters and log files, and the default plots are pooled in a folder in the current working directory. Thus, in its default configuration, *MLAir* performs a meaningful meteorological ML experiment, which can serve as a benchmark for further developments and baseline for more sophisticated ML architectures.

In the second example (Fig. 3), we enlarged the `window_history_size` (number of previous time steps) of the input data to provide more contextual information to the vanilla model. Furthermore, we use a different set of observational stations as indicated in the parameter `stations`. From a first glance, the output of the experiment run is quite similar to the earlier exam-

```python
import mlair

# just give it a dry run without any modification
mlair.run()
```

```
INFO: DefaultWorkflow started
INFO: ExperimentSetup started
INFO: Experiment path is: /home/<usr>/mlair/testrun_network
...
INFO: load data for DEBW107 from JOIN
INFO: load data for DEBY081 from JOIN
INFO: load data for DEBW013 from JOIN
INFO: load data for DEBW076 from JOIN
INFO: load data for DEBW087 from JOIN
...
INFO: Training started
...
INFO: DefaultWorkflow finished after 0:03:04 (hh:mm:ss)
```

Figure 2. A very simple *Python* script (e.g. written in a *Jupyter Notebook* (Kluyver et al., 2016) or *Python* file) calling the *MLAir* package without any modification. Selected parts of the corresponding logging of the running code are shown underneath. Results of this and following code snippets have to be seen as a pure demonstration, because the default neural network is very simple.

```python
import mlair

# our new stations to use
stations = ['DEBW030', 'DEBW037', 'DEBW031', 'DEBW015', 'DEBW107']

# expanded temporal context to 14 (days, because of default sampling="daily")
window_history_size = 14

# restart the experiment with little customisation
mlair.run(stations=stations,
          window_history_size=window_history_size)
```

```
INFO: DefaultWorkflow started
INFO: ExperimentSetup started
...
INFO: load data for DEBW030 from JOIN
INFO: load data for DEBW037 from JOIN
INFO: load data for DEBW031 from JOIN
INFO: load data for DEBW015 from JOIN
...
INFO: Training started
...
INFO: DefaultWorkflow finished after 00:02:03 (hh:mm:ss)
```

Figure 3. The *MLAir* experiment has now minor adjustments for the parameters `stations` and `window_history_size`.

ple. However, there are a couple of aspects in this second experiment which we would like to point out. Firstly, the `DefaultDataHandler` keeps track of data available locally and thus reduces the overhead of reloading data from the web if this is not necessary. Therefore, no new data were downloaded for station `DEBW107`, which is part of the default configuration, as its data have already been stored locally in our first experiment. Of course the `DefaultDataHandler` can be forced to reload all data from their source if needed (see Sect. 4.1). The second key aspect to highlight here is that the parameter `window_history_size` could be changed, and the network was trained anew without any problem even though this change affects the shape of the input data and thus the neural network architecture. This is possible because the model class in *MLAir* queries the shape of the input variables and adapts the architecture of the input layer accordingly. Naturally, this procedure does not make perfect sense for every model, as it only affects the first layer of the model. In case the shape of the input data changes drastically, it is advisable to adapt the entire model as well. Concerning the network output, the second experiment overwrites all results from the first run, because without an explicit setting of the file path, *MLAir* always uses the same sandbox directory called `testrun_network`. In a real-world sequence of experiments, we recommend always specifying a new experiment path with a reasonably descriptive name (details on the experiment path in Sect. 4.1).

The third example in this section demonstrates the activation of a partial workflow, namely a re-evaluation of a previously trained neural network. We want to rerun the evaluation part with a different set of stations to perform an independent validation. This partial workflow is also employed if the model is run in production. As we replace the stations for the new evaluation, we need to create a new testing set, but we want to skip the model creation and training steps. Hence, the parameters `create_new_model` and `train_model` are set to `False` (see Fig. 4). With this setup, the model is loaded from the local file path and the evaluation is performed on the newly provided stations. By combining the stations from the second and third experiment in the `stations` parameter the model could be evaluated at all selected stations together. In this setting, *MLAir* will abort to execute the evaluation if parameters pertinent for preprocessing or model compilation changed compared to the training run.

It is also possible to continue training of an already trained model. If the `train_model` parameter is set to `True`, training will be resumed at the last epoch reached previously, if this epoch number is lower than the `epochs` parameter. Specific uses for this are either an experiment interruption (for example due to wall clock time limit exceedance on batch systems) or the desire to extend the training if the optimal network weights have not been found yet. Further details on training resumption can be found in Sect. 4.9.

3.2 Results of an experiment

All results of an experiment are stored in the directory, which is defined during the experiment setup stage (see Sect. 4.1). The sub-directory structure is created at the beginning of the experiment. There is no automatic deletion of temporary files in case of aborted runs so that the information that is generated up to the program termination can be inspected to find potential errors or to check on a successful initialization of the model, etc. Figure 5 shows the output file structure. The content of each directory is as follows:

- All samples used for training and validation are stored in the `batch_data` folder.
- `forecasts` contains the actual predictions of the trained model and the persistence and linear references. All forecasts (model and references) are provided in normalized and original value ranges. Additionally, the optional bootstrap forecasts are stored here (see Sect. 3.3).
- In `latex_report`, there are publication-ready tables in *Markdown* (Gruber, 2004) or *LaTeX* (LaTeX Project, 2005) format, which give a summary about the stations used, the number of samples, and the hyperparameters and experiment settings.
- The `logging` folder contains information about the execution of the experiment. In addition to the console output, *MLAir* also stores messages on the debugging level, which give a better understanding of the internal program sequence. *MLAir* has a tracking functionality, which can be used to trace which data have been stored and pulled from the central data store. In combination with the corresponding tracking plot that is created at the very end of each experiment automatically, it allows visual tracking of which parameters have an effect on which stage. This functionality is most interesting for developers who make modifications to the source code and want to ensure that their changes do not break the data flow.
- The folder `model` contains everything that is related to the trained model. Besides the file, which contains the model itself (stored in the binary hierarchical data format *HDF5*; Koranne, 2011), there is also an overview graphic of the model architecture and all *Keras* callbacks, for example from the learning rate. If a training is not started from the beginning but is either continued or applied to a pre-trained model, all necessary information like the model or required callbacks must be stored in this subfolder.
- The `plots` directory contains all graphics that are created during an experiment. Which graphics are to be created in postprocessing can be determined using the `plot_list` parameter in the experiment setup. In addition, *MLAir* automatically generates monitoring plots, for instance of the evolution of the loss during training.

As described in the last bullet point, all plots which are created during an *MLAir* experiment can be found in the subfolder `plots`. By default, all available plot types are created. By explicitly naming individual graphics in the `plot_list` parameter, it is possible to override this behaviour and specify which graphics are created during postprocessing. Additional plots are created to monitor the training behaviour. These graphics are always created when a training session is

```python
import mlair

# our new stations to use
stations = ['DEBY002', 'DEBY079']

# same setting for window_history_size
window_history_size = 14

# run experiment without training
mlair.run(stations=stations,
          window_history_size=window_history_size,
          create_new_model=False,
          train_model=False)
```

```
INFO: DefaultWorkflow started
...
INFO: No training has started, because train_model parameter was false.
...
INFO: DefaultWorkflow finished after 0:01:27 (hh:mm:ss)
```

Figure 4. Experiment run without training. For this, it is required to have an already trained model in the experiment path.

carried out. Most of the plots which are created in the course of postprocessing are publication-ready graphics with complete legend and resolution of 500 dpi. Custom graphics can be added to *MLAir* by attaching an additional run module (see Sect. 4.8) which contains the graphic creation methods.

A general overview of the underlying data can be obtained with the graphics `PlotStationMap` and `PlotAvailability`. `PlotStationMap` (Fig. 6) marks the geographical position of the stations used on a plain map with a land–sea mask, country boundaries, and major water bodies. The data availability chart created by `PlotAvailability` (Fig. 7) indicates the time periods for which preprocessed data for each measuring station are available. The lowest bar shows whether a station with measurements is available at all for a certain point in time. The three subsets of training, validation, and testing data are highlighted in different colours.

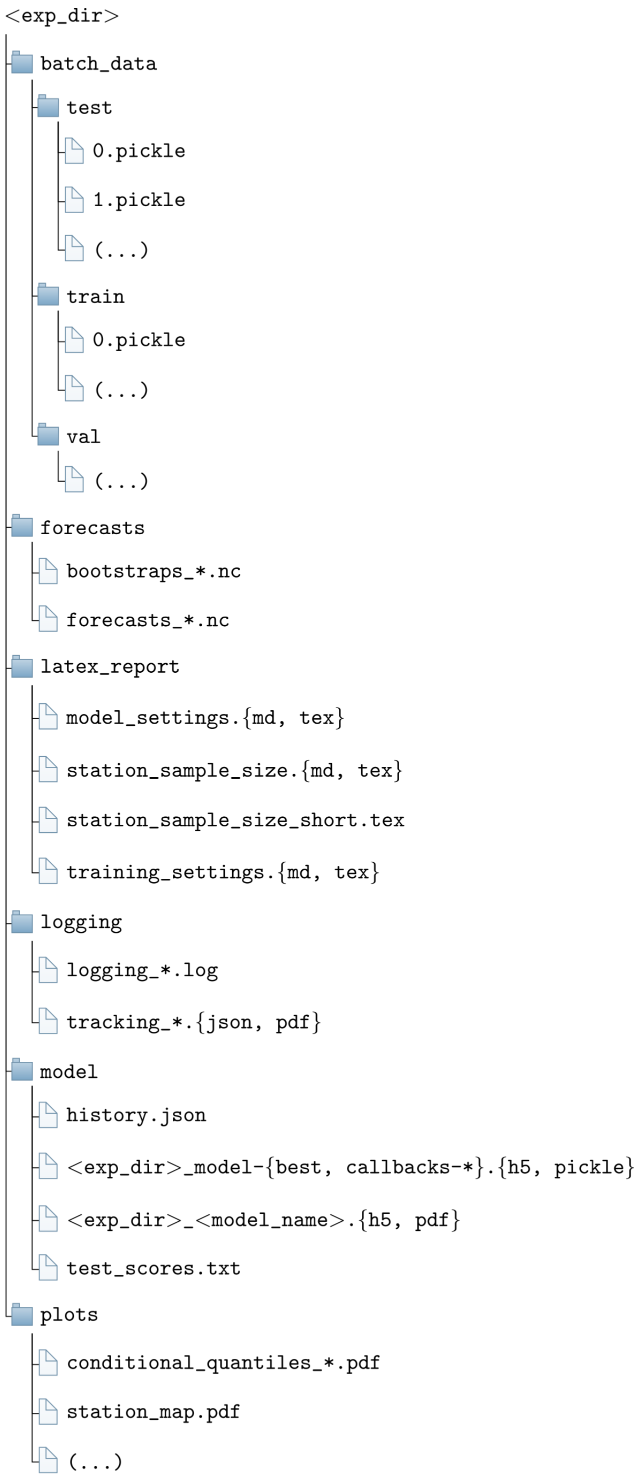
The monitoring graphics show the course of the loss function as well as the error depending on the epoch for the training and validation data (see Fig. 8). In addition, the error of the best model state with respect to the validation data is shown in the plot title. If the learning rate is modified during the course of the experiment, another plot is created to show its development. These monitoring graphics are kept as simple as possible and are meant to provide insight into the training process. The underlying data are always stored in the *JavaScript Object Notation* format (.json, ISO Central Secretary, 2017) in the subfolder `model` and can therefore be used for case-specific analyses and plots.

Through the graphs `PlotMonthlySummary` and `PlotTimeSeries` it is possible to quickly assess the forecast quality of the ML model. The `PlotMonthlySummary` (see Fig. 9) summarizes all predictions of the model covering all stations but considering each month separately as a box-and-whisker diagram. With this graph it is possible to get a general overview of the distribution of the predicted values compared to the distribution of the observed values for each month. Besides, the exact course of the time series compared to the observation can be viewed in the `PlotTimeSeries` (not included as a figure in this article). However, since this plot has to scale according to the length of the time series, it should be noted that this last-mentioned graph is kept very simple and is generally not suitable for publication.

3.3 Statistical analysis of results

A central element of *MLAir* is the statistical evaluation of the results according to state-of-the-art methods used in meteorology. To obtain specific information on the forecasting model, we treat forecasts and observations as random variables. Therefore, the joint distribution $p(m,o)$ of a model m and an observation o contains information on $p(m)$, $p(o)$ (marginal distribution), and the relations $p(o|m)$ and $p(m|o)$ (conditional distribution) between both of them (Murphy and Winkler, 1987). Following Murphy et al. (1989), marginal distribution is shown as a histogram (light grey), while the conditional distribution is shown as percentiles in different line styles. By using `PlotConditionalQuantiles`, *MLAir* automatically creates plots for the entire test period (Fig. 10) that are, as is common in meteorology, separated by seasons.

In order to access the genuine added value of a new forecasting model, it is essential to take other existing forecasting models into account instead of reporting only metrics related to the observation. In *MLAir* we implemented three types of basic reference forecasts: (i) a persistence forecast, (ii) an or-



```
<exp_dir>
├ batch_data
│ ├ test
│ │ ├ 0.pickle
│ │ ├ 1.pickle
│ │ └ (...)
│ ├ train
│ │ ├ 0.pickle
│ │ └ (...)
│ └ val
│   └ (...)
├ forecasts
│ ├ bootstraps_*.nc
│ └ forecasts_*.nc
├ latex_report
│ ├ model_settings.{md, tex}
│ ├ station_sample_size.{md, tex}
│ ├ station_sample_size_short.tex
│ └ training_settings.{md, tex}
├ logging
│ ├ logging_*.log
│ └ tracking_*.{json, pdf}
├ model
│ ├ history.json
│ ├ <exp_dir>_model-{best, callbacks-*}.{h5, pickle}
│ ├ <exp_dir>_<model_name>.{h5, pdf}
│ └ test_scores.txt
└ plots
  ├ conditional_quantiles_*.pdf
  ├ station_map.pdf
  └ (...)
```

Figure 5. Default structure of each *MLAir* experiment with the subfolders `forecasts`, `latex_report`, `logging`, `model`, and `plots`. `<exp_dir>` is a placeholder for the actual name of the experiment.

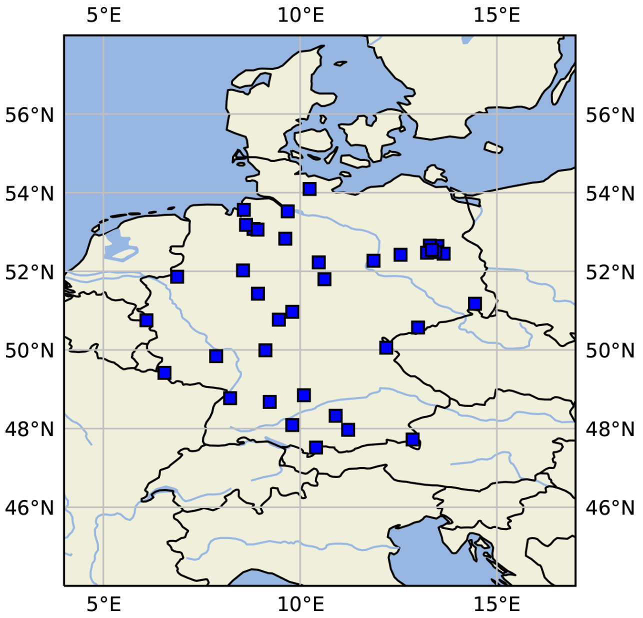


Figure 6. Map of central Europe showing the locations of some sample measurement stations as blue squares created by `PlotStationMap`.

dinary multi-linear least square model, and (iii) four climatological forecasts.

The persistence forecast is based on the last observed time step, which is then used as a prediction for all lead times. The ordinary multi-linear least square model serves as a linear competitor and is derived from the same data the model was trained with. For the climatological references, we follow Murphy (1988) who defined single and multiple valued climatological references based on different timescales. We refer the reader to Murphy (1988) for an in-depth discussion of the climatological reference. Note that this kind of persistence and also the climatological forecast might not be applicable for all temporal resolutions and may therefore need adjustment in different experiment settings. We think here, for example, of a clear diurnal pattern in temperature, for which a persistence of successive observations would not provide a good forecast. In this context, a reference forecast based on the observation of the previous day at the same time might be more suitable.

For the comparison, we use a skill score S, which is naturally defined as the performance of a new forecast compared to a competitive reference with respect to a statistical metric (Murphy and Daan, 1985). Applying the mean squared error as the statistical metric, such a skill score S reduces to unity minus the ratio of the error of the forecast to the reference. A positive skill score can be interpreted as the percentage of improvement of the new model forecast in comparison to the reference. On the other hand, a negative skill score denotes that the forecast of interest is less accurate than the referenc-

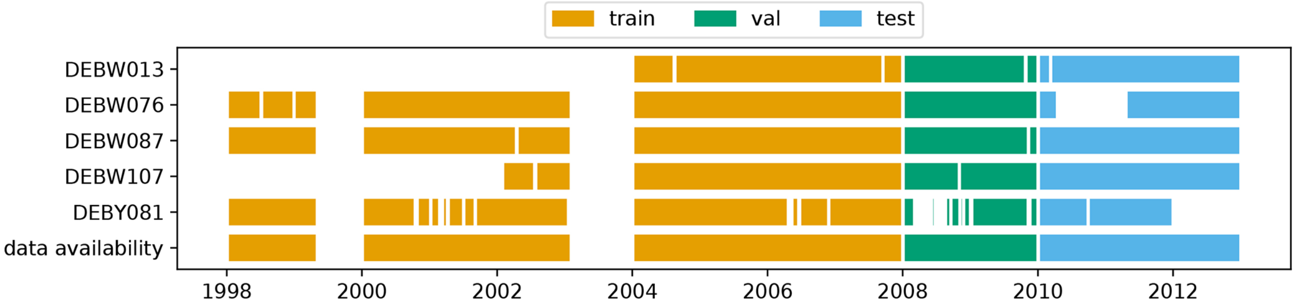


Figure 7. `PlotAvailability` diagram showing the available data for five measurement stations. The different colours denote which period of the time series is used for the training (orange), validation (green), and test (blue) data set. “Data availability” denotes if any of the above-mentioned stations has a data record for a given time.

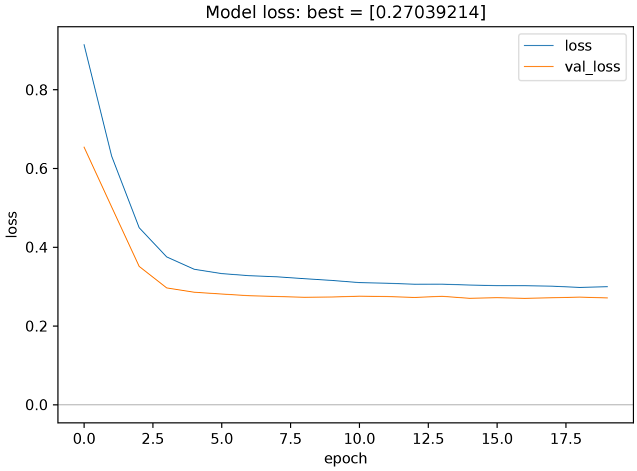


Figure 8. Monitoring plots showing the evolution of train and validation loss as a function of the number of epochs. This plot type is kept very simplistic by choice. The underlying data are saved during the experiment so that it would be easy to create a more advanced plot using the same data.

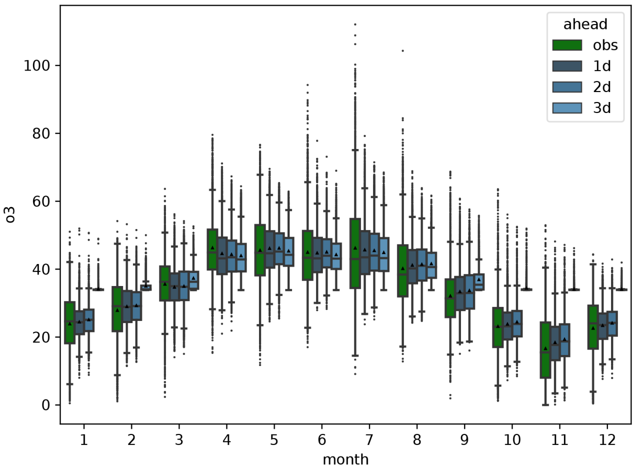


Figure 9. Graph of `PlotMonthlySummary` showing the observations (green) and the predictions for all forecast steps (dark to light blue) separated for each month.

ing forecast. Consequently, a value of zero denotes that both forecasts perform equally (Murphy, 1988).

The `PlotCompetitiveSkillScore` (Fig. 11) includes the comparison between the trained model, the persistence, and the ordinary multi-linear least squared regression. The climatological skill scores are calculated separately for each forecast step (lead time) and summarized as a box-and-whiskers plot over all stations and forecasts (Fig. 12), and as a simplified version showing the skill score only (not shown) using `PlotClimatologicalSkillScore`.

In addition to the statistical model evaluation, *MLAir* also allows the importance of individual input variables to be assessed through bootstrapping of individual input variables. For this, the time series of each individual input variable is resampled n times (with replacement) and then fed to the trained network. By resampling a single input variable, its temporal information is disturbed, but the general frequency distribution is preserved. The latter is important because it ensures that the model is provided only with values from a known range and does not extrapolate out-of-sample. Afterwards, the skill scores of the bootstrapped predictions are calculated using the original forecast as reference. Input variables that show an overly negative skill score during bootstrapping have a stronger influence on the prediction than input variables with a small negative skill score. In case the bootstrapped skill score even reaches the positive value domain, this could be an indication that the examined variable has no influence on the prediction at all. The result of this approach applied to all input variables is presented in `PlotBootstrapSkillScore` (Fig. 13). A more detailed description of this approach is given in Kleinert et al. (2021).

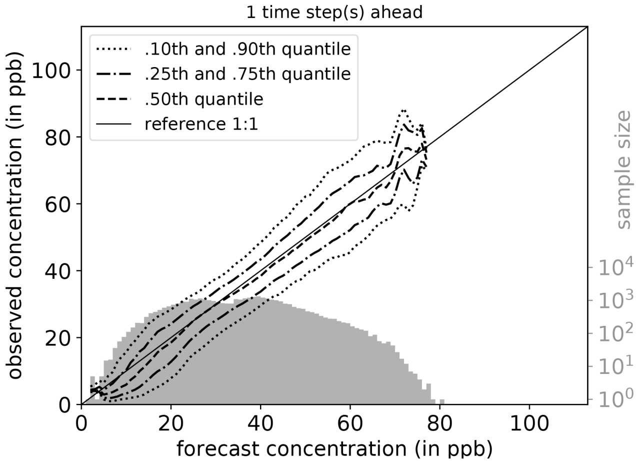


Figure 10. Conditional quantiles in terms of calibration-refinement factorization for the first lead time and the full test period. The marginal forecasting distribution is shown as a log histogram in light grey (counting on right axis). The conditional distribution (calibration) is shown as percentiles in different line styles. Calculations are done with a bin size of 1 ppb. Moreover, the percentiles are smoothed by a rolling mean of window size three. This kind of plot was originally proposed by Murphy et al. (1989) and can be created using `PlotConditionalQuantiles`.

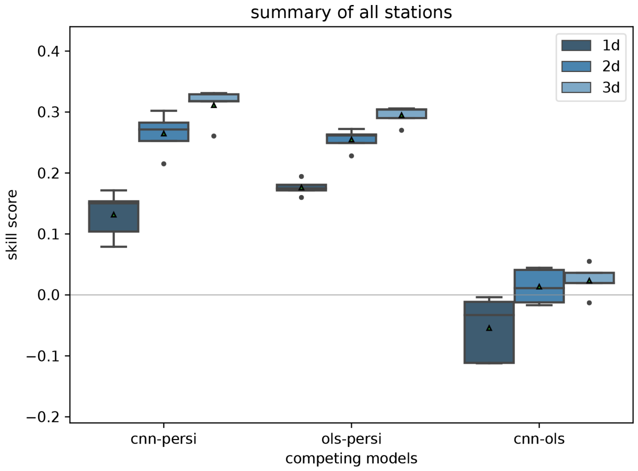


Figure 11. Skill scores of different reference models like persistence (persi) and ordinary multi-linear least square (ols). Skill scores are shown separately for all forecast steps (dark to light blue). This graph is generated by invoking `PlotCompetitiveSkillScore`.

4 Configuration of experiment, data handler, and model class in the MLAir workflow

As well as the already described workflow adjustments, *MLAir* offers a large number of configuration options. Instead of defining parameters at different locations inside the code, all parameters are centrally set in the experiment setup. In this section, we describe all parameters that can be modified and the authors' choices for default settings when using the default workflow of *MLAir*.

4.1 Host system and processing units

The *MLAir* workflow can be adjusted to the hosting system. For that, the local paths for experiment and data are adjustable (see Table 1 for all options). Both paths are separated by choice. This has the advantage that the same data can be used multiple times for different experiment setups if stored outside the experiment path. Contrary to the data path placement, all created plots and forecasts are saved in the `experiment_path` by default, but this can be adjusted through the `plot_path` and `forecast_path` parameter.

Concerning the processing units, *MLAir* supports both central processing units (CPUs) and GPUs. Due to their bandwidth optimization and efficiency on matrix operations, GPUs have become popular for ML applications (see Krizhevsky et al., 2012). Currently, the sample models implemented in *MLAir* are based on *TensorFlow* v1.13.1, which has distinct branches: the *tensorflow-1.13.1* package for CPU computation and the *tensorflow-gpu-1.13.1* package for GPU devices. Depending on the operating system, the user needs to install the appropriate library if using *TensorFlow* releases 1.15 and older (TensorFlow, 2020). Apart from this installation issue, *MLAir* is able to detect and handle both *TensorFlow* versions during run time. An *MLAir* version to support *TensorFlow* v2 is planned for the future (see Sect. 5).

4.2 Preprocessing

In the course of preprocessing, the data are prepared to allow immediate use in training and evaluation without further preparation. In addition to the general data acquisition and formatting, which will be discussed in Sect. 4.3 and 4.4, preprocessing also handles the split into training, validation, and test data. All parameters discussed in this section are listed in Table 2.

Data are split into subsets along the temporal axis and station between a hold-out data set (called test data) and the data that are used for training (training data) and model tuning (validation data). For each subset, a `{train,val,test}_start` and `{train,val,test}_end` date not exceeding the overall time span (see Sect. 4.4) can be set. Additionally, for each subset it is possible to define a minimal number of samples per station `{train,val,test}_min_length` to remove very short time series that potentially cause misleading results especially in the validation and test phase. A spatial split of the data is achieved by assigning each station to one of the three subsets of data. The parameter `fraction_of_training` determines the ratio between hold-out data and data for training and validation, where the latter two are always split with a ratio of 80 % to 20 %, which is a typical choice for these subsets.

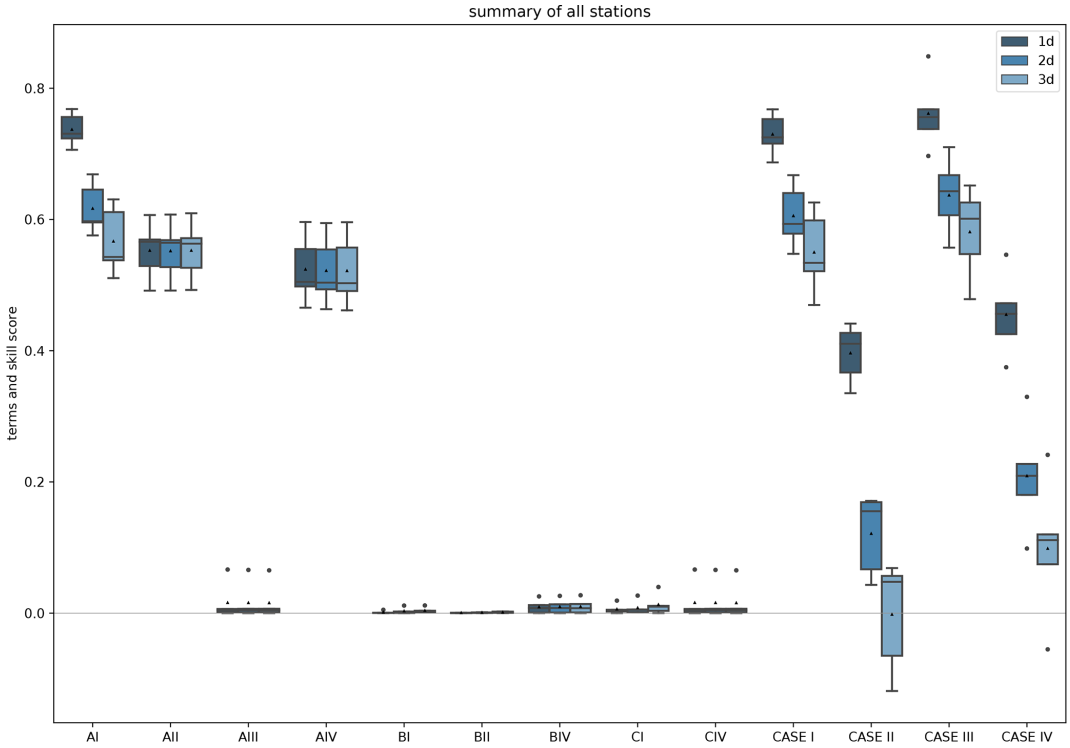


Figure 12. Climatological skill scores (cases I to IV) and related terms of the decomposition as proposed in Murphy (1988) created by `PlotClimatologicalSkillScore`. Skill scores and terms are shown separately for all forecast steps (dark to light blue). In brief, cases I to IV describe a comparison with climatological reference values evaluated on the test data. Case I is the comparison of the forecast with a single mean value formed on the training and validation data and case II with the (multi-value) monthly mean. The climatological references for cases III and IV are, analogous to cases I and II, the single and the multi-value mean, but on the test data. Cases I to IV are calculated from the terms AI to CIV. For more detailed explanations of the cases, we refer to Murphy (1988).

Table 1. Summary of all parameters related to the host system that are required, recommended, or optional to adjust for a custom experiment workflow.

Host system		
Parameter	Default	Adjustment
`experiment_date`	testrun	recommended
`experiment_name`	`{experiment_date}`_network	–[a]
`experiment_path`	⟨cwd[b]⟩`/{experiment_name}`	optional
`data_path`	⟨cwd[b]⟩`/data`	optional
`bootstrap_path`	⟨`data_path`⟩`/bootstraps`	optional
`forecast_path`	⟨`experiment_path`⟩`/forecasts`	optional
`plot_path`	⟨`experiment_path`⟩`/plots`	optional

[a] Only adjustable via the `experiment_date` parameter.
[b] Refers to the Linux command to get the path name of the current working directory.

To achieve absolute statistical data subset independence, data should ideally be split along both temporal and spatial dimensions. Since the spatial dependency of two distinct stations may vary due to weather regimes, season, and time of day (Wilks, 2011), a spatial and temporal division of the data might be useful, as otherwise a trained model can presumably lead to over-confident results. On the other hand, by applying a spatial split in combination with a temporal division, the amount of utilizable data can drop massively. In *MLAir*, it is therefore up to the user to split data either in the temporal dimension or along both dimensions by using the `use_all_stations_on_all_data_sets` parameter.

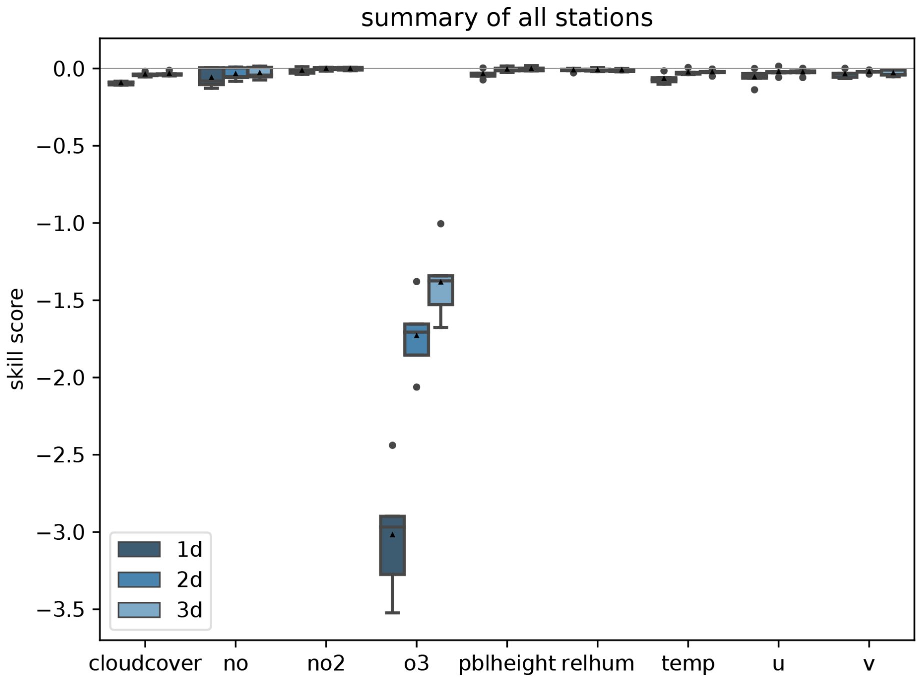


Figure 13. Skill score of bootstrapped model input predictions separated for each input variable (x axis) and forecast steps (dark to light blue) with the original (non-bootstrapped) predictions as reference. `PlotBootstrapSkillScore` is only executed if bootstrap analysis is enabled.

Table 2. Summary of all parameters related to the preprocessing that are required, recommended, or optional to adjust for a custom experiment workflow.

Preprocessing		
Parameter	Default	Adjustment
`stations`	default stations[a]	recommended
`data_handler`	`DefaultDataHandler`	optional
`fraction_of_training`	0.8	optional[b]
`use_all_stations_on_all_data_sets`	True	optional

[a] Default stations: DEBW107, DEBY081, DEBW013, DEBW076, DEBW087.
[b] Not used in the default setup because `use_all_stations_on_all_data_sets` is True.

4.3 Custom data handler

The integration of a custom data handler into the *MLAir* workflow is done by inheritance from the `AbstractDataHandler` class and implementation of at least the constructor `__init__()`, and the accessors `get_X()`, and `get_Y()`. The custom data handler is added to the *MLAir* workflow as a parameter without initialization. At runtime, *MLAir* then queries all the required parameters of this custom data handler from its arguments and keyword arguments, loads them from the data store and finally calls the constructor. If data need to be downloaded or preprocessed, this should be executed inside the constructor. It is sufficient to load the data in the accessor methods if the data can be used without conversion. Note that a data handler is only responsible for preparing data from a single origin, while the iteration and distribution into batches is taken care of while *MLAir* is running.

The accessor methods for input and target data form a clearly defined interface between *MLAir*'s run modules and the custom data handler. During training the data are needed as a *NumPy* array; for preprocessing and evaluation the data are partly used as *xarray*. Therefore the accessor methods have the parameter `as_numpy` and should be able to return both formats. Furthermore it is possible to use a custom upsampling technique for training. To activate this feature the parameter `upsampling` can be enabled. If such a technique is not used and therefore not implemented, the parameter has no further effect.

The abstract data handler provides two additional placeholder methods that can support data preparation, training, and validation. Depending on the case, it may be helpful to

Table 3. Summary of all parameters related to the default data handler that are required, recommended, or optional to adjust for a custom experiment workflow.

Default data handler		
Parameter	Default	Adjustment
`data_path`	see Table 1	optional
`stations`	default stations[a]	recommended
`network`	–	optional
`station_type`	–	optional
`variables`	default variables[b]	recommended
`statistics_per_var`	default statistics[b]	recommended
`target_var`	o3	recommended
`start`	1997-01-01	recommended
`end`	2017-12-31	recommended
`sampling`	daily	optional
`window_history_size`	13	recommended
`interpolation_method`	linear	optional
`limit_nan_fill`	1	optional
`min_length`[c]	0	optional
`window_lead_time`	3	recommended
`overwrite_local_data`	False	optional

[a] Default stations: DEBW107, DEBY081, DEBW013, DEBW076, DEBW087.
[b] Default variables (statistics): o3 (dma8eu), relhum (average_values), temp (maximum), *u* (average_values), *v* (average_values), no (dma8eu), no2 (dma8eu), cloudcover (average_values), pblheight (maximum).
[c] Indicates the required minimum number of samples per station.

define these methods within a custom data handler. With the method `transformation` it is possible to either define or calculate the transformation properties of the data handler before initialization. The returned properties are then applied to all subdata sets, namely training, validation, and testing. Another supporting class method is `get_coordinates`. This method is currently used only for the map plot for geographical overview (see Sect. 3.2). To feed the overview map, this method must return a dictionary with the geographical coordinates indicated by the keys `lat` and `lon`.

4.4 Default data handler

In this section we describe a concrete implementation of a data handler, namely the `DefaultDataHandler`, using data from the JOIN interface.

Regarding the data handling and preprocessing, several parameters can be set to control the choice of inputs, size of data, etc. in the data handler (see Table 3). First, the underlying raw data must be downloaded from the web. The current version of the `DefaultDataHandler` is configured for use with the REST API of the JOIN interface (Schultz and Schröder, 2017). Alternatively, data could be already available on the local machine in the directory `data_path`, e.g. from a previous experiment run. Additionally, a user can force *MLAir* to load fresh data from the web by enabling the `overwrite_local_data` parameter. According to the design structure of a data handler, data are handled separately for each observational station indicated by its ID. By default, the `DefaultDataHandler` uses all German air quality stations provided by the German Environment Agency (Umweltbundesamt, UBA) that are indicated as "background" stations according to the European Environmental Agency (EEA) AirBase classification (European Parliament and Council of the European Union, 2008). Using the `stations` parameter, a user-defined data collection can be created. To filter the stations, the parameters `network` and `station_type` can be used as described in Schultz et al. (2017a) and the documentation of JOIN (Schultz and Schröder, 2017).

For the `DefaultDataHandler`, it is recommended to specify at least

- the number of preceding time steps to use for a single input sample (`window_history_size`),
- if and which interpolation should be used (`interpolation_method`),
- if and how many missing values are allowed to be filled by interpolation (`limit_nan_fill`),
- and how many time steps the forecast model should predict (`window_lead_time`).

Regarding the data content itself, each requested variable must be added to the `variables` list and be part of the `statistics_per_var` dictionary together with a proper statistic abbreviation (see documentation of Schultz and Schröder, 2017). If not provided, both parameters are chosen from a standard set of variables and statistics. Similar actions are required for the target variable. Firstly, target variables are defined in `target_var`, and secondly, the target variable must also be part of the `statistics_per_var` parameter. Note that the JOIN REST API calculates these statistics online from hourly values, thereby taking into account a minimum data coverage criterion. Finally, the overall time span the data shall cover can be defined via `start` and `end`, and the temporal resolution of the data is set by a string like `"daily"` passed to the `sampling` parameter. At this point, we want to refer to Sect. 5, where we discuss the temporal resolution currently available.

4.5 Defining a model class

The motivation behind using model classes was already explained in Sect. 2.4. Here, we show more details on the implementation and customization.

To achieve the goal of an easy plug-and-play behaviour, each ML model implemented in *MLAir* must inherit from the `AbstractModelClass`, and the methods `set_model` and `set_compile_options` are required to be overwritten for the custom model. Inside `set_model`, the entire model from inputs to outputs is created. Thereby it has to be ensured that the model is compatible with *Keras* to be compiled. *MLAir* supports both the functional and sequential *Keras* application programming interfaces. For details on how to create a model with *Keras*, we refer to the official *Keras* documentation (Chollet et al., 2015). All options for the model compilation should be set in the `set_compile_options` method. This method should at least include information on the training algorithm (`optimizer`), and the loss to measure performance during training and optimize the model for (`loss`). Users can add other compile options like the learning rate (`learning_rate`), `metrics` to report additional informative performance metrics, or options regarding the weighting as `loss_weights`, `sample_weight_mode`, or `weighted_metrics`. Finally, methods that are not part of *Keras* or *TensorFlow* like customized loss functions or self-made model extensions are required to be added as so-called `custom_objects` to the model so that *Keras* can properly use these custom objects. For that, it is necessary to call the `set_custom_objects` method with all custom objects as key value pairs. See also the official *Keras* documentation for further information on custom objects.

An example implementation of a small model using a single convolution and three fully connected layers is shown in Fig. 14. By inheriting from the `AbstractModelClass` (l. 9), invoking its constructor (l. 15), defining the methods `set_model` (l. 25–35) and `set_compile_options` (l. 37–41), and calling these two methods (l. 21–22), the custom model is immediately usable for *MLAir*. Additionally, the loss is added to the custom objects (l. 23). This last step would not be necessary in this case, because an error function incorporated in *Keras* is used (l. 2/40). For the purpose of demonstrating how to use a customized loss, it is added nevertheless.

A more elaborate example is described in Kleinert et al. (2021), who used extensions to the standard *Keras* library in their workflow. So-called inception blocks (Szegedy et al., 2015) and a modification of the two-dimensional padding layers were implemented as *Keras* layers and could be used in the model afterwards.

4.6 Training

The parameter `create_new_model` instructs *MLAir* to create a new model and use it in the training. This is necessary, for example, for the very first training run in a new experiment. However, it must be noted that already existing training progress within the experiment will be overwritten by activating `create_new_model`. Independent of using a new or already existing model, `train_model` can be used to set whether the model is to be trained or not. Further notes on the continuation of an already started training or the use of a pre-trained model can be found in Sect. 4.9.

Most parameters to set for the training stage are related to hyperparameter tuning (see Table 4). Firstly, the `batch_size` can be set. Furthermore, the number of `epochs` to train needs to be adjusted. Last but not least, the `model` used itself must be provided to *MLAir* including additional hyperparameters like the `learning_rate`, the algorithm to train the model (`optimizer`), and the `loss` function to measure model performance. For more details on how to implement an ML model properly we refer to Sect. 4.5.

Due to its application focus on meteorological time series and therefore on solving a regression problem, *MLAir* offers a particular handling of training data. A popular technique in ML, especially in the image recognition field, is to augment and randomly shuffle data to produce a larger number of input samples with a broader variety. This method requires independent and identically distributed data. For meteorological applications, these techniques cannot be applied out of the box, because of the lack of statistical independence of most data and autocorrelation (see also Schultz et al., 2021). To avoid generating over-confident forecasts, training and test data are split into blocks so that little or no overlap remains between the data sets. Another common problem in ML, not only in the meteorological context, is the natural under-representation of extreme values, i.e. an imbalance problem. To address this issue, *MLAir* allows more emphasis to be placed on such data points. The weighting of data samples is conducted by an over-representation of values that can be considered as extreme regarding the deviation from a mean state in the output space. This can be applied during training by using the `extreme_values` parameter, which defines a threshold value at which a value is considered extreme. Training samples with target values that exceed this limit are then used a second time in each epoch. It is also possible to enter more than one value for the parameter. In this case, samples with values that exceed several limits are duplicated according to the number of limits exceeded. For positively skewed distributions, it could be helpful to apply this over-representation only on the right tail of the distribution (`extremes_on_right_tail_only`). Furthermore, it is possible to shuffle data within, and only within, the training subset randomly by enabling `permute_data`.

4.7 Validation

The configuration of the ML model validation is related to the postprocessing stage. As mentioned in Sect. 2.3, in the default configuration there are three major validation steps undertaken after each run besides the creation of graphics: first, the trained model is opposed to the two reference models, a simple linear regression and a persistence prediction. Second, these models are compared with climatological statistics. Lastly, the influence of each input variable is estimated by a bootstrap procedure.

Due to the computational burden the calculation of the input variable sensitivity can be skipped and the graphics creation part can be shortened. To perform the sensitivity study, the parameter `evaluate_bootstraps` must be enabled and the `number_of_bootstraps` defines how many

```python
import keras
from keras.losses import mean_squared_error as mse
from keras.optimizers import SGD

from mlair.model_modules import AbstractModelClass

from mlair.workflows import DefaultWorkflow

class MyCustomisedModel(AbstractModelClass):

    """
    A customised model with a 1x1 Conv, and 2 Dense layers (16,
    output shape). Dropout is used after Conv layer.
    """
    def __init__(self, input_shape: list, output_shape: list):

        # set attributes _input_shape and _output_shape
        super().__init__(input_shape[0], output_shape[0])

        # apply to model
        self.set_model()
        self.set_compile_options()
        self.set_custom_objects(loss=self.compile_options['loss'])

    def set_model(self):
        x_input = keras.layers.Input(shape=self._input_shape)
        x_in = keras.layers.Conv2D(32, (1, 1))(x_input)
        x_in = keras.layers.PReLU()(x_in)
        x_in = keras.layers.Flatten()(x_in)
        x_in = keras.layers.Dropout(0.1)(x_in)
        x_in = keras.layers.Dense(16)(x_in)
        x_in = keras.layers.PReLU()(x_in)
        x_in = keras.layers.Dense(self._output_shape)(x_in)
        out = keras.layers.PReLU()(x_in)
        self.model = keras.Model(inputs=x_input, outputs=[out])

    def set_compile_options(self):
        self.initial_lr = 1e-2
        self.optimizer = SGD(lr=self.initial_lr, momentum=0.9)
        self.loss = mse
        self.compile_options = {"metrics": ["mse", "mae"]}

# Make use of MyCustomisedModel within the DefaultWorkflow
workflow = DefaultWorkflow(model=MyCustomisedModel, epochs=2)
workflow.run()
```

Figure 14. Example how to create a custom ML model implemented as a model class. MyCustomisedModel has a single 1 × 1 convolution layer followed by two fully connected layers with a neuron size of 16, and the number of forecast steps. The model itself is defined in the set_model method, whereas compile options such as the optimizer, loss, and error metrics are defined in set_compile_options. Additionally, for demonstration, the loss is added as custom object which is not required because a *Keras* built-in function is used as loss.

samples shall be drawn for the evaluation (see Table 5). If such a sensitivity study was already performed and the training stage was skipped, the create_new_bootstraps parameter should be set to False to reuse already preprocessed samples if possible. To control the creation of graphics, the parameter plot_list can be adjusted. If not specified, a default selection of graphics is generated. When using plot_list, each graphic to be drawn must be specified individually. More details about all possible graphics have already been provided in Sect. 3.2 and 3.3. In the current version, extending the validation as part of *MLAir*'s default postprocessing stage is somewhat complicated, but it is possible to append another run module to the workflow to perform additional validations.

Table 4. Summary of all parameters related to the training that are required, recommended, or optional to adjust for a custom experiment workflow.

Training		
Parameter	Default	Adjustment
`train_model`	False	recommended[a]
`create_new_model`	False	recommended[a]
`batch_size`	512	optional
`epochs`	20	optional
`loss`[b]	–	required
`metrics`[b]	–	optional
`model`	vanilla model[c]	required
`learning_rate`[b]	–	required
`optimizer`[b]	–	required
`extreme_values`	–	optional
`extremes_on_right_tail_only`	False	optional
`permute_data`	False	optional

[a] Note: both parameters are disabled per default to prevent unintended overwriting of a model. If, upon reversion, these parameters are not enabled on the first execution of a new experiment without providing a suitable and trained ML model, the *MLAir* workflow is going to fail.
[b] These parameters are set in the model class.
[c] As default, a vanilla feed-forward neural network architecture will be loaded for workflow testing. The usage of such a simple network for a real application is at least questionable.

Table 5. Summary of all parameters related to the evaluation that are required, recommended, or optional to adjust for a custom experiment workflow.

Evaluation		
Parameter	Default	Adjustment
`plot_list`	default plots[a]	optional
`evaluate_bootstraps`	True	optional
`number_of_bootstraps`	20	optional
`create_new_bootstraps`	False[b]	optional

[a] Default plots are `PlotMonthlySummary`, `PlotStationMap`, `PlotClimatologicalSkillScore`, `PlotTimeSeries`, `PlotCompetitiveSkillScore`, `PlotBootstrapSkillScore`, `PlotConditionalQuantiles`, and `PlotAvailability`.
[b] Is automatically enabled if parameter `train_model` (see Table 4) is enabled.

4.8 Custom run modules and workflow adaptions

MLAir offers the possibility to define and execute a custom workflow for situations in which special calculations or data evaluation procedures not available in the standard version are needed. For this purpose it is not necessary to modify the program code of *MLAir*, but instead user-defined run modules can be included in a new workflow. This is done in analogy to the procedure of defining new model classes by inheritance from the base class `RunEnvironment`. Compared to the very simple examples from Sect. 3, such a use of *MLAir* requires a slightly increased effort. The implementation of the run module is done straightforwardly by a constructor method, which initializes the module and executes all desired calculation steps when called. To execute the custom workflow, the *MLAir* `Workflow` class must be loaded and then each run module must be registered. The order in which the individual stages are added determines the execution sequence.

As custom workflows will generally be necessary if a custom run module is to be defined, we briefly describe how the central data store mentioned in Sect. 2.3 interacts with the workflow module. With the data store it is possible to share any kind of information from previous or subsequent stages. By invoking the constructor of the super class during the initialization of a custom run module, the data store is automatically connected with this module. Information can then be set or queried using the accessor methods `get` and `set`. For each saved information object a separate namespace called `scope` can be assigned. If not specified, the object is always stored in the general scope. If the scope is specified, a separate sub-scope is created. Information stored in this scope memory cannot be accessed from the general scope memory, but conversely all sub-scopes have access to the general scope. For example, more general objects can be set in the general scope and objects specific to a sub-data set, such as test data, can be stored under the scope `test`. If some objects for the keyword `test` are retrieved from the data store, then for non-existent objects in the `test` namespace attributes from the general scope are used if available.

An example for the implementation of a custom run module embedded in a custom workflow can be found in Fig. 15. The custom run module named `CustomStage` inherits from the base class `RunEnvironment` (l. 4) and calls its constructor (l. 8) on initialization. The `CustomStage` expects a single parameter (`test_string`, l. 7), which

```
 1  import mlair
 2  import logging
 3
 4  class CustomStage(mlair.RunEnvironment):
 5      """A custom MLAir stage for demonstration."""
 6
 7      def __init__(self, test_string):
 8          super().__init__()  # always call super init method
 9          self._run(test_string)  # call a class method
10
11      def _run(self, test_string):
12          logging.info("Just running a custom stage.")
13          logging.info("test_string = " + test_string)
14          epochs = self.data_store.get("epochs")
15          logging.info("epochs = " + str(epochs))
16
17
18  # create your custom MLAir workflow
19  CustomWorkflow = mlair.Workflow()
20  # provide stages without initialisation
21  CustomWorkflow.add(mlair.ExperimentSetup, epochs=128)
22  # add also keyword arguments for a specific stage
23  CustomWorkflow.add(CustomStage, test_string="Hello World")
24  # finally execute custom workflow in order of adding
25  CustomWorkflow.run()
```

```
INFO: Workflow started
...
INFO: ExperimentSetup finished after 00:00:12 (hh:mm:ss)
INFO: CustomStage started
INFO: Just running a custom stage.
INFO: test_string = Hello World
INFO: epochs = 128
INFO: CustomStage finished after 00:00:01 (hh:mm:ss)
INFO: Workflow finished after 00:00:13 (hh:mm:ss)
```

Figure 15. Embedding of a custom run module in a modified *MLAir* workflow. In comparison to Figs. 2, 3, and 4, this code example works on a single step deeper regarding the level of abstraction. Instead of calling the run method of *MLAir*, the user needs to add all stages individually and is responsible for all dependencies between the stages. By using the `Workflow` class as context manager, all stages are automatically connected with the result that all stages can easily be plugged in.

is used during the `run` method (l. 11–15). The `run` method first logs two information messages by using the `test_string` parameter (l. 12–13). Then it extracts the value of the parameter `epochs` (l. 14) that has been set in the `ExperimentSetup` (l. 21) from the data store and logs the value of this parameter too. To run this custom run module is has to be included in a workflow. First an empty workflow is created (l. 19) and then individual run modules are attached (l. 21–23). As last step, this new defined workflow is executed by calling the `run` method (l. 25).

4.9 How to continue an experiment?

There can be different reasons for the continuation of an experiment. First of all, by looking at the monitoring graphs, it could be discovered that training has not yet converged and the number of epochs should be increased. Instead of training a new network from scratch, the training can be resumed from the latest epoch to save time. To do so, the parameter `epochs` must be increased accordingly and `create_new_model` must be set to `False`. If the `model` output folder has not been touched, the intermediate results and the history of the previous training are usually available in full, so that *MLAir* can continue the training as if it had never been interrupted. Another reason for a continuation would be the interruption of the training for unexpected reasons such as runtime exceedance on batch systems. By keeping the same number of epochs and switching off the creation of a new model, the training continues at the last checkpoint (see model setup in Sect. 2.3). Finally, *MLAir* can also be used in the context of transfer learning. By providing a pre-trained model and having `train_model` enabled and create_new_model disabled, a transfer learning task can be performed.

5 Limitations

Even though *MLAir* addresses a wide range of ML-related problems and allows many different ML architectures and customized workflows to be embedded, it is still no universal Swiss Army knife but rather focuses on the application of neural networks for the task of station time series forecasting. In this section we will explain the limitations of *MLAir* and why *MLAir* ends at these points.

Due to the scientifically oriented development of *MLAir* starting from a specific research question (Kleinert et al., 2021), *MLAir* could initially only use data from the REST API of JOIN. This binding has already been revoked in the current version, however, the DefaultDataHandler still uses this data source. Furthermore, *MLAir* always expects a particular structure in the data and especially considers the data as a collection of time series data from various stations. We are currently investigating the possibility of integrating grid data, which could be taken from a weather model, and time-constant data such as topography into the *MLAir* workflow, but we cannot yet present any results on how easy such an integration would be.

While *MLAir* can technically handle data in different time resolutions, it has been tested primarily on daily aggregated data due to the specific science case which served as the seed for its development. The use of different temporal resolutions was spot-checked and could be successfully confirmed without obvious errors, but we cannot guarantee that the results will be meaningful if data in other temporal resolutions are used as inputs. In particular, most of the evaluation routines may not make sense for data in less than hourly or greater than daily resolution. Note also that *MLAir* does not perform explicit error checking or missing value handling. Such functionality must be implemented within the data handler. *MLAir* expects a ready-to-use data set without missing values provided by the data handler during training.

Another limitation is the choice of the underlying libraries and their versions. Due to the selection of *TensorFlow* as back-end, it is not possible to use *PyTorch* or other frameworks in combination with *MLAir*. Specifically, *MLAir* was developed and tested with *TensorFlow* version 1.13.1, as the HPC systems on which our experiments are performed supported this version at the time of writing. We have already tested *MLAir* occasionally with the *TensorFlow* version 1.15 and could not find any errors. Please check the code repository for updates concerning the support of newer *TensorFlow* versions, which we hope to make available in the coming months.

6 Summary

MLAir is an innovative software package intended to facilitate high-quality meteorological studies using ML. By providing an end-to-end solution based on a specific scientific workflow of time series prediction, *MLAir* enables a transparent and reproducible conduction of ML experiments in this domain. Due to the plug-and-play behaviour it is straightforward to explore different model architectures and change various aspects of the workflow or model evaluation. Although *MLAir* is focusing on neural networks, it should be possible to include other ML techniques. Since *MLAir* is based on a pure *Python* environment, and it is highly portable. It has been tested on various computing systems from desktop workstations to high-end supercomputers.

MLAir is under continuous development. Further enhancements of the program are already planned and can be found in the issue tracker (see Code availability). Ongoing developments concern the extension of the statistical evaluation methods, the graphical presentation of the results, and the flawless support of temporal resolutions other than daily aggregated data. Through further code refactoring, *MLAir* will become even more versatile as the decoupling of individual components is being pushed forward. In particular, it is planned to structure the data handling in a more modular way so that varying structured data sources can be connected and used without much effort. We invite the community of meteorological ML scientists to participate in the further development of *MLAir* through comments and contributions to code and documentation. A good starting point for contributions is the issue tracker of *MLAir*.

We hope that *MLAir* can serve as a blueprint for the development of reusable ML applications in the fields of meteorology and air quality, as it seeks to combine the best practices from ML with the best practices of meteorological model evaluation and data preprocessing. *MLAir* is thus a contribution to strengthen cooperation between the communities of ML and meteorology or air quality researchers.

Code availability. The current version of *MLAir* is available from the project website https://gitlab.version.fz-juelich.de/toar/mlair (last access: 10 March 2021) under the MIT licence. The exact version v1.0.0 of *MLAir* described in this paper and used to produce the code examples shown is archived on *B2SHARE* (Leufen et al., 2020, https://doi.org/10.34730/5a6c3533512541a79c5c41061743f5e3). Detailed installation instructions are provided in the project page readme file. There is also a *Jupyter notebook* with all code snippets to reproduce the examples highlighted in this paper.

Data availability. *MLAir* is not directly linked to any specific data. Data used in the examples are extracted from the corresponding databases at runtime, as described and cited in the text.

Author contributions. The concept of MLAir was developed by LHL. Detailed investigations on methodological approaches were carried out by LHL and FK. FK contributed the first use case. The actual implementation of the software and its validation and visualization were done by LHL and FK. LHL was responsible for the code structure and test developments. MGS contributed to the concept discussions and provided supervision and funding. LHL wrote the original draft, all authors reviewed the manuscript and helped in the preparation of the final paper as well as drafting the responses to the reviewers' comments.

Competing interests. The authors declare that they have no conflict of interest.

Acknowledgements. The authors gratefully acknowledge the Jülich Supercomputing Centre for providing the infrastructure and resources to develop this software.

Financial support. This research has been supported by the European Research Council, H2020 Research Infrastructures (IntelliAQ (grant no. 787576).

The article processing charges for this open-access
publication were covered by a Research
Centre of the Helmholtz Association.

Review statement. This paper was edited by Christoph Knote and reviewed by two anonymous referees.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, available at: https://www.tensorflow.org/ (last access: 10 March 2021), 2015.

Bentayeb, M., Wagner, V., Stempfelet, M., Zins, M., Goldberg, M., Pascal, M., Larrieu, S., Beaudeau, P., Cassadou, S., Eilstein, D., Filleul, L., Le Tertre, A., Medina, S., Pascal, L., Prouvost, H., Quénel, P., Zeghnoun, A., and Lefranc, A.: Association between long-term exposure to air pollution and mortality in France: a 25-year follow-up study, Environ. Int., 85, 5–14, https://doi.org/10.1016/j.envint.2015.08.006, 2015.

Bishop, C. M.: Pattern recognition and machine learning, Springer, New York, 2006.

Brunner, D., Savage, N., Jorba, O., Eder, B., Giordano, L., Badia, A., Balzarini, A., Baró, R., Bianconi, R., Chemel, C., Curci, G., Forkel, R., Jiménez-Guerrero, P., Hirtl, M., Hodzic, A., Honzak, L., Im, U., Knote, C., Makar, P., Manders-Groot, A., van Meijgaard, E., Neal, L., Pérez, J. L., Pirovano, G., San Jose, R., Schröder, W., Sokhi, R. S., Syrakov, D., Torian, A., Tuccella, P., Werhahn, J., Wolke, R., Yahya, K., Zabkar, R., Zhang, Y., Hogrefe, C., and Galmarini, S.: Comparative analysis of meteorological performance of coupled chemistry-meteorology models in the context of AQMEII phase 2, Atmos. Environ., 115, 470–498, https://doi.org/10.1016/j.atmosenv.2014.12.032, 2015.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y.: Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, arXiv: 1406.1078, available at: http://arxiv.org/abs/1406.1078 (last access: 10 March 2021), 2014.

Chollet, F., et al.: Keras, available at: https://keras.io (last access: 10 March 2021), 2015.

Cohen, A. J., Anderson, H. R., Ostro, B., Pandey, K. D., Krzyzanowski, M., Künzli, N., Gutschmidt, K., Pope, A., Romieu, I., Samet, J. M., and Smith, K.: The Global Burden of Disease Due to Outdoor Air Pollution, J. Toxicol. Env. Hea. A, 68, 1301–1307, https://doi.org/10.1080/15287390590936166, 2005.

Elliott, T.: The State of the Octoverse: machine learning, The GitHub Blog, available at: https://github.blog/2019-01-24-the-state-of-the-octoverse-machine-learning/, (last access: 23 June 2020), 2019.

European Parliament and Council of the European Union: Directive 2008/50/EC of the European Parliament and of the Council of 21 May 2008 on ambient air quality and cleaner air for Europe, Official Journal of the European Union, available at: http://data.europa.eu/eli/dir/2008/50/oj (last access: 10 March 2021), 2008.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y.: Generative Adversarial Nets, in: Advances in Neural Information Processing Systems 27, edited by: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q., pp. 2672–2680, Curran Associates, Inc., available at: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf (last access: 10 March 2021), 2014.

Goodfellow, I., Bengio, Y., and Courville, A.: Deep Learning, MIT Press, http://www.deeplearningbook.org (last access: 10 March 2021), 2016.

Gruber, J.: Markdown, available at: https://daringfireball.net/projects/markdown/license, (last access: 7 January 2021), 2004.

Hochreiter, S. and Schmidhuber, J.: Long Short-Term Memory, Neural Computation, 9, 1735–1780, https://doi.org/10.1162/neco.1997.9.8.1735, 1997.

Hoyer, S. and Hamman, J.: xarray: N-D labeled arrays and datasets in Python, J. Open Res. Softw., 5, 10, https://doi.org/10.5334/jors.148, 2017.

Hoyer, S., Hamman, J., Roos, M., Fitzgerald, C., Cherian, D., Fujii, K., Maussion, F., crusaderky, Kleeman, A., Kluyver, T., Clark, S., Munroe, J., keewis, Hatfield-Dodds, Z., Nicholas, T., Abernathey, R., Wolfram, P. J., MaximilianR, Hauser, M., Markel, Gundersen, G., Signell, J., Helmus, J. J., Sinai, Y. B., Cable, P., Amici, A., lumbric, Rocklin, M., Rivera, G., and Barna, A.: pydata/xarray v0.15.0, Zenodo, https://doi.org/10.5281/zenodo.3631851, 2020.

ISO Central Secretary: Information technology – The JSON data interchange syntax, Standard ISO/IEC 21778:2017, International

Organization for Standardization, Geneva, Switzerland, available at: https://www.iso.org/standard/71616.html (last access: 10 March 2021), 2017.

Kingma, D. P. and Welling, M.: Auto-Encoding Variational Bayes, arXiv: 1312.6114, available at: https://arxiv.org/abs/1312.6114 (last access: 10 March 2021), 2014.

Kleinert, F., Leufen, L. H., and Schultz, M. G.: IntelliO3-ts v1.0: a neural network approach to predict near-surface ozone concentrations in Germany, Geosci. Model Dev., 14, 1–25, https://doi.org/10.5194/gmd-14-1-2021, 2021.

Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., Willing, C., and development team, J.: Jupyter Notebooks – a publishing format for reproducible computational workflows, in: Positioning and Power in Academic Publishing: Players, Agents and Agendas, edited by: Loizides, F. and Scmidt, B., IOS Press, the Netherlands, 87–90, available at: https://eprints.soton.ac.uk/403913/ (last access: 10 March 2021), 2016.

Koranne, S.: Hierarchical data format 5: HDF5, in: Handbook of Open Source Tools, 191–200, Springer, Boston, MA, HDF5 is maintained by The HDF Group, http://www.hdfgroup.org/HDF5 (last access: 10 March 2021), 2011.

Krizhevsky, A., Sutskever, I., and Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks, in: Advances in Neural Information Processing Systems 25, edited by: Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q., Curran Associates, Inc., 1106–1114, available at: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks (last access: 10 March 2021), 2012.

LaTeX Project: LaTeX, available at: https://www.latex-project.org/, (last access: 7 January 2021), 2005.

Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P.: Gradient-based learning applied to document recognition, Proceedings of the IEEE, 86, 2278–2324, https://doi.org/10.1109/5.726791, 1998.

Lefohn, A. S., Malley, C. S., Smith, L., Wells, B., Hazucha, M., Simon, H., Naik, V., Mills, G., Schultz, M. G., Paoletti, E., De Marco, A., Xu, X., Zhang, L., Wang, T., Neufeld, H. S., Musselman, R. C., Tarasick, D., Brauer, M., Feng, Z., Tang, H., Kobayashi, K., Sicard, P., Solberg, S., and Gerosa, G.: Tropospheric ozone assessment report: Global ozone metrics for climate change, human health, and crop/ecosystem research, Elementa: Science of the Anthropocene, 1, 1, https://doi.org/10.1525/elementa.279, 2018.

Leufen, L. H., Kleinert, F., and Schultz, M. G.: MLAir (v1.0.0) – a tool to enable fast and flexible machine learning on air data time series – Source Code, EUDAT Collaborative Data Infrastructure, https://doi.org/10.34730/fcc6b509d5394dad8cfdfc6e9fff2bec, 2020.

Mills, G., Pleijel, H., Malley, C., Sinha, B., Cooper, O., Schultz, M., Neufeld, H., Simpson, D., Sharps, K., Feng, Z., Gerosa, G., Harmens, H., Kobayashi, K., Saxena, P., Paoletti, E., Sinha, V., and Xu, X.: Tropospheric Ozone Assessment Report: Present-day tropospheric ozone distribution and trends relevant to vegetation, Elementa: Science of the Anthropocene, 6, 47, https://doi.org/10.1525/elementa.302, 2018.

Murphy, A. H.: Skill Scores Based on the Mean Square Error and Their Relationships to the Correlation Coefficient, Mon. Weather Rev., 116, 2417–2424, https://doi.org/10.1175/1520-0493(1988)116<2417:SSBOTM>2.0.CO;2, 1988.

Murphy, A. H. and Daan, H.: Forecast evaluation, in: Probability, statistics, and decision making in the atmospheric sciences, edited by: Murphy, A. H. and Katz, R. W., Westview Press, Boulder, USA, 379–437, 1985.

Murphy, A. H. and Winkler, R. L.: A General Framework for Forecast Verification, Mon. Weather Rev., 115, 1330–1338, https://doi.org/10.1175/1520-0493(1987)115<1330:AGFFFV>2.0.CO;2, 1987.

Murphy, A. H., Brown, B. G., and Chen, Y.-S.: Diagnostic Verification of Temperature Forecasts, Weather Forecast., 4, 485–501, https://doi.org/10.1175/1520-0434(1989)004<0485:DVOTF>2.0.CO;2, 1989.

Musgrave, K., Belongie, S., and Lim, S.-N.: A Metric Learning Reality Check, arXiv: 2003.08505, available at: https://arxiv.org/abs/2003.08505 (last access: 10 March 2021), 2020.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S.: PyTorch: An Imperative Style, High-Performance Deep Learning Library, in: Advances in Neural Information Processing Systems 32, edited by: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., Curran Associates, Inc., Vancouver, Canada, 8024–8035, available at: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf (last access: 10 March 2021), 2019.

Python Software Foundation: Python Language Reference, release 3.6.8, PEP 494, available at: https://www.python.org/dev/peps/pep-0494/ (last access: 10 March 2021), 2018.

Reback, J., McKinney, W., jbrockmendel, Van den Bossche, J., Augspurger, T., Cloud, P., gfyoung, Sinhrks, Klein, A., Roeschke, M., Tratner, J., She, C., Hawkins, S., Ayd, W., Petersen, T., Schendel, J., Hayden, A., Garcia, M., MomIsBestFriend, Jancauskas, V., Battiston, P., Seabold, S., chris-b1, h-vetinari, Hoyer, S., Overmeire, W., alimcmaster1, Mehyar, M., Dong, K., and Whelan, C.: pandas-dev/pandas: Pandas v1.0.1, Zenodo, https://doi.org/10.5281/zenodo.3644238, 2020.

Rezende, D. J., Mohamed, S., and Wierstra, D.: Stochastic Backpropagation and Approximate Inference in Deep Generative Models, arXiv: 1401.4082, available at: https://arxiv.org/abs/1401.4082 (last access: 10 March 2021), 2014.

Schultz, M. G. and Schröder, S.: Documentation of the JOIN REST interface, Juelich, Germany, available at: https://join.fz-juelich.de/services/rest/surfacedata/, (last access: 18 September 2020), 2017.

Schultz, M. G., Schröder, S., Lyapina, O., Cooper, O. R., Galbally, I., Petropavlovskikh, I., von Schneidemesser, E., Tanimoto, H., Elshorbany, Y., Naja, M., Seguel, R. J., Dauert, U., Eckhardt, P., Feigenspan, S., Fiebig, M., Hjellbrekke, A.-G., Hong, Y.-D., Kjeld, P. C., Koide, H., Lear, G., Tarasick, D., Ueno, M., Wallasch, M., Baumgardner, D., Chuang, M.-T., Gillett, R., Lee, M., Molloy, S., Moolla, R., Wang, T., Sharps, K., Adame, J. A., Ancellet, G., Apadula, F., Artaxo, P., Barlasina, M. E., Bogucka, M., Bonasoni, P., Chang, L., Colomb, A., Cuevas-Agulló, E., Cupeiro, M., Degorska, A., Ding, A., Fröhlich, M., Frolova, M., Gadhavi, H., Gheusi, F., Gilge, S., Gonzalez, M. Y., Gros, V., Hamad, S. H., Helmig, D., Henriques, D., Hermansen, O., Holla, R., Hueber, J., Im, U., Jaffe, D. A., Komala, N., Kubistin, D., Lam, K.-S., Laurila, T., Lee, H., Levy, I., Mazzoleni, C., Mazzoleni, L. R., McClure-Begley, A., Mohamad, M., Murovec, M., Navarro-Comas, M., Nicodim, F., Parrish, D., Read, K. A., Reid, N., Ries, L., Saxena, P., Schwab, J. J., Scorgie, Y., Senik, I., Simmonds, P., Sinha, V., Skorokhod, A. I., Spain, G., Spangl, W., Spoor, R., Springston, S. R., Steer, K., Steinbacher, M., Suharguniyawan, E., Torre, P., Trickl, T., Weili, L., Weller, R., Xiaobin, X., Xue, L., and Zhiqiang, M.: Tropospheric Ozone Assessment Report: Database and metrics data of global surface ozone observations, Elementa: Science of the Anthropocene, 5, 58, https://doi.org/10.1525/elementa.244, 2017a.

Schultz, M. G., Schröder, S.,Lyapina, O., Cooper, O. R., Galbally, I., Petropavlovskikh, I., von Schneidemesser, E., Tanimoto, H., Elshorbany, Y., Naja, M., Seguel, R. J., Dauert, U., Eckhardt, P., Feigenspan, S., Fiebig, M., Hjellbrekke, A.-G., Hong, Y.-D., Kjeld, P. C., Koide, H., Lear, G., Tarasick, D., Ueno, M., Wallasch, M., Baumgardner, D., Chuang, M.-T., Gillett, R., Lee, M., Molloy, S., Moolla, R., Wang, T., Sharps, K., Adame, J. A., Ancellet, G.., Apadula, F., Artaxo, P., Barlasina, M. E., Bogucka, M., Bonasoni, P., Chang, L., Colomb, A., Cuevas-Agulló, E., Cupeiro, M., Degorska, A., Ding, A., Fröhlich, M., Frolova, M., Gadhavi, H., Gheusi, F., Gilge, S., Gonzalez, M. Y., Gros, V., Hamad, S. H., Helmig, D., Henriques, D., Hermansen, O., Holla, R., Hueber, J., Im, U., Jaffe, D. A., Komala, N., Kubistin, D., Lam, K.-S., Laurila, T., Lee, H.,Levy, I., Mazzoleni, C., Mazzoleni, L. R., McClure-Begley, A., Mohamad, M., Murovec, M., Navarro-Comas, M., Nicodim, F., Parrish, D., Read, K. A., Reid, N., Ries, L., Saxena, P., Schwab, J. J.,Scorgie, Y., Senik, I., Simmonds, P., Sinha, V., Skorokhod, A. I., Spain, G., Spangl, W., Spoor, R., Springston, S. R., Steer, K., Steinbacher, M., Suharguniyawan, E., Torre, P., Trickl, T., Weili, L., Weller, R., Xu, X., Xue, L., and Zhiqiang, M.: Tropospheric Ozone Assessment Report, links to Global surface ozone datasets, PANGAEA, https://doi.org/10.1594/PANGAEA.876108, 2017b.

Schultz, M. G., Betancourt, C., Gong, B., Kleinert, F., Langguth, M., Leufen, L. H., Mozaffari, A., and Stadtler, S.: Can deep learning beat numerical weather prediction?, Philos. T. Roy. Soc. A, 379, 2194, https://doi.org/10.1098/rsta.2020.0097, 2021.

Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A.: Going deeper with convolutions, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, Boston, MA, USA, 1–9, https://doi.org/10.1109/CVPR.2015.7298594, 2015.

TensorFlow: GPU support, available at: https://www.tensorflow.org/install/gpu, last access: 6 June 2020.

TOAR: Tropospheric Ozone Assessment Report (TOAR): Global metrics for climate change, human health and crop/ecosystem research, International Global Atmospheric Chemistry (IGAC), available at: https://igacproject.org/activities/TOAR (last access: 29 January 2021), 2014–2021.

US Environmental Protection Agency: Integrated science assessment for ozone and related photochemical oxidants, US Environmental Protection Agency, Washington, D.C., ePA-HQ-ORD-2018-0274, 2020.

van der Walt, S., Colbert, S. C., and Varoquaux, G.: The NumPy Array: A Structure for Efficient Numerical Computation, Comput. Sci. Eng., 13, 22–30, https://doi.org/10.1109/MCSE.2011.37, 2011.

Vautard, R.: Evaluation of the meteorological forcing used for the Air Quality Model Evaluation International Initiative (AQMEII) air quality simulations, Atmos. Environ., 53, 15–37, https://doi.org/10.1016/j.atmosenv.2011.10.065, 2012.

Wes McKinney: Data Structures for Statistical Computing in Python, in: Proceedings of the 9th Python in Science Conference, edited by: Stéfan van der Walt and Jarrod Millman, SciPy Organizers, Austin, Texas, 56–61, https://doi.org/10.25080/Majora-92bf1922-00a, 2010.

Wilks, D. S. (Ed.): Statistical methods in the atmospheric sciences, pp. 178–186, International Geophysics Series, Elsevier Academic Press, Amsterdam, 3rd edn., 2011.

World Health Organization: Health risks of air pollution in Europe – HRAPIE project recommendations for concentration–response functions for cost–benefit analysis of particulate matter, ozone and nitrogen dioxide, Ozone and Nitrogen Dioxide, available at: https://www.euro.who.int/__data/assets/pdf_file/0006/238956/Health_risks_air_pollution_HRAPIE_project.pdf (last access: 10 March 2021), 2013.

D.2. Leufen et al. (2022): *Exploring decomposition of temporal patterns to facilitate learning of neural networks for ground-level daily maximum 8-hour average ozone prediction*

Environmental Data Science (2022), 1: e10, 1–31
doi:10.1017/eds.2022.9

CAMBRIDGE
UNIVERSITY PRESS

APPLICATION PAPER

Exploring decomposition of temporal patterns to facilitate learning of neural networks for ground-level daily maximum 8-hour average ozone prediction

Lukas Hubert Leufen[1,2,*], Felix Kleinert[1,2] and Martin G. Schultz[1]

[1]Jülich Supercomputing Centre, Forschungszentrum Jülich, Jülich, Germany
[2]Institute of Geosciences, University of Bonn, Bonn, Germany
*Corresponding author. Email: l.leufen@fz-juelich.de

Received: 15 November 2021; **Revised:** 05 May 2022; **Accepted:** 31 May 2022

Keywords: Air quality; deep learning; machine learning; ozone; temporal decomposition; time series prediction

Abstract

Exposure to ground-level ozone is a concern for both humans and vegetation, so accurate prediction of ozone time series is of great importance. However, conventional as well as emerging methods have deficiencies in predicting time series when a superposition of differently pronounced oscillations on various time scales is present. In this paper, we propose a meteorologically motivated filtering method of time series data, which can separate oscillation patterns, in combination with different multibranch neural networks. To avoid phase shifts introduced by using a causal filter, we combine past observation data with a climatological estimate about the future to be able to apply a noncausal filter in a forecast setting. In addition, the forecast in the form of the expected climatology provides some a priori information that can support the neural network to focus not merely on learning a climatological statistic. We apply this method to hourly data obtained from over 50 different monitoring stations in northern Germany situated in rural or suburban surroundings to generate a prediction for the daily maximum 8-hr average values of ground-level ozone 4 days into the future. The data preprocessing with time filters enables simpler neural networks such as fully connected networks as well as more sophisticated approaches such as convolutional and recurrent neural networks to better recognize long-term and short-term oscillation patterns like the seasonal cycle and thus leads to an improvement in the forecast skill, especially for a lead time of more than 48 hr, compared to persistence, climatological reference, and other reference models.

Impact Statement

Exposure to ground-level ozone harms humans and vegetation, but the prediction of ozone time series, especially by machine learning, encounters problems due to the superposition of different oscillation patterns from long-term to short-term scales. Decomposing the input time series into long-term and short-term signals with the help of climatology and statistical filtering techniques can improve the prediction of various neural network architectures due to an improved recognition of different temporal patterns. More reliable and accurate forecasts support decision-makers and individuals in taking timely and necessary countermeasures to air pollution episodes.

This research article was awarded Open Data and OpenMaterials badges for transparent practices. See the Data Availability Statement for details.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

1. Introduction

Human health and vegetation growth are impaired by ground-level ozone (REVIHAAP, 2013; US EPA, 2013; Monks et al., 2015; Maas and Grennfelt, 2016; Fleming et al., 2018). High short-term ozone exposures cause worsening of symptoms, a need for stronger medication, and an increase in emergency hospital admissions, for people with asthma or chronic obstructive pulmonary diseases in particular (US EPA, 2020). More broadly, ozone exposure also increases susceptibility to respiratory diseases such as pneumonia in general, which in turn leads to an increased likelihood of hospitalization (US EPA, 2020). Findings of Di et al. (2017) further support earlier research that short-term exposure to ozone, even below regulatory limits, is highly likely to increase the risk of premature death, particularly for the elderly. Since the 1990s, there have been major changes in the global distribution of anthropogenic emissions (Richter et al., 2005; Granier et al., 2011; Russell et al., 2012; Hilboll et al., 2013; Zhang et al., 2016), which in turn has an influence on the ozone concentrations. Although reductions in peak concentrations have been achieved (Simon et al., 2015; Lefohn et al., 2017; Fleming et al., 2018), the negative effects of ground-level ozone remain (Cohen et al., 2017; Seltzer et al., 2017; Zhang et al., 2018; Shindell et al., 2019). Recent studies show that within the European Union, for example, ozone has the greatest impact on highly industrialized countries such as Germany, France, or Spain (Ortiz and Guerreiro, 2020). For all these reasons, it is therefore of utmost importance to be able to predict ozone as accurately as possible in the short term.

In light of these impacts, it is desirable to accurately forecast ozone concentrations for a couple of days so that protection measures can be initiated in time. Chemical transport models (CTMs), which explicitly solve the underlying chemical and physical equations, are commonly used to predict ozone (e.g., Collins et al., 1997; Wang et al., 1998a, 1998b; Horowitz et al., 2003; von Kuhlmann et al., 2003; Grell et al., 2005; Donner et al., 2011). Even though CTMs are equipped with the most up-to-date knowledge of research, the resulting estimates for exposure to and impacts of ozone may vary enormously between different CTM studies (Seltzer et al., 2020). Since CTMs operate on a computational grid and are thus always dependent on simplification of processes, parameterizations, and further assumptions, CTMs are themselves affected by large uncertainties (Manders et al., 2012). The deviations in the output of CTMs result accordingly from chemical and physical processes, fluxes such as emissions or deposition, as well as meteorological phenomena (Vautard et al., 2012; Bessagnet et al., 2016; Young et al., 2018). Finally, in order to use the predictions of the CTMs at the level of measuring stations, either model output statistics have to be applied (Fuentes and Raftery, 2005) or statistical methods are required (Lou Thompson et al., 2001).

In addition to simpler methods such as multilinear regressions, statistical methods that can map the relationship between time and observations in the time series are also suitable for this purpose. In general, time series can be characterized by the fact that values that are close in time tend to be similar or correlated (Wilks, 2006) and that the temporal ordering of these values forms an essential property of the time series (Bagnall et al., 2017). Autoregressive models (ARs) use this relationship and calculate the next value of a series x_{i+1} as a function ϕ of past values $x_i, x_{i-1}, \ldots, x_{i-n}$ where ϕ is simply a linear regression. Autoregressive moving average models extend this approach by additionally considering the error of past values, which is not described by the AR model. In the case of nonstationary time series, autoregressive integrated moving average models are used. However, these approaches are mostly limited to univariate problems and can only represent linear relationships (Shih et al., 2019). Alternative developments of nonlinear statistical models, such as Monte Carlo simulations or bootstrapping methods, have therefore been used for nonlinear predictions (De Gooijer and Hyndman, 2006).

In times of high availability of large data and increasingly efficient computing systems, machine learning (ML) has become an excellent alternative to classical statistical methods (Reichstein et al., 2019). ML is a generic term for data-driven algorithms like decision trees, random forests, or neural networks (NNs), which usually determine their parameters in a data-hungry and time-consuming learning process and can then be applied to new data at relatively low cost in terms of time and computational effort.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Fully connected networks (FCNs) are the pioneers of NNs and were already successfully applied around the turn of the millennium, for example, for the prediction of meteorological and air quality problems (Comrie, 1997; Gardner and Dorling, 1999; Elkamel et al., 2001; Kolehmainen et al., 2001). Simply put, FCNs extend the classical method of multilinear regression by adding the properties of nonlinearity as well as learning of knowledge. From a theoretical point of view, a sufficiently large network can be assumed to be a universal approximator of any function (Hornik et al., 1989). Nevertheless, it also shows that the application of FCNs is limited because they ignore the topology of the inputs (LeCun et al., 1999). In terms of time series, this means that FCNs will not be able to understand the abstract concept of unidirectional time.

These shortcomings have been overcome to some extent by deep learning (DL). In general, any NN that has a more sophisticated architecture or is based on more than three layers is classified as a deep NN. As Schultz et al. (2021) describe, the history of DL has been marked by highs and lows, as both computational cost and the size of datasets have always been tough adversaries. Since the 2010s, DL's more recent advances can be attributed to three main points: First, the acquisition of new knowledge has been drastically accelerated by massive parallel computation using graphics processing units. Second, so-called convolutional neural networks (CNNs; LeCun et al., 1999) became popular, whose strength lies in their ability to contextualize individual data points better than previous neural networks by sharing weights within the network and thus learning more information while maintaining the same network size. Finally, due to ever-increasing digitization, more and more data are available in ever-improving quality. Since DL methods are purely data-based compared to classical statistics, greater knowledge can be built up within a neural network simply through the greater availability of data.

Various newer NN architectures have been developed and also applied to time series forecasting in recent years. In this study, we focus on CNNs and recurrent neural networks (RNNs) as competitors to an FCN. For the prediction of time series, CNNs offer an advantage over FCNs due to their ability to better map relationships between neighboring data points. In Earth sciences, time series are typically multivariate, since a single time series is rarely considered in isolation, but always in interaction with other variables. However, multivariate time series should not be treated straightforwardly as two-dimensional images, since a causal relationship between different time series does not necessarily exist at all times and a different order of these time series would influence the result. Multivariate time series are therefore better to be understood as a composite of different one-dimensional data series (Zheng et al., 2014). Following this fact, multivariate time series can best be considered as a one-dimensional picture with different color channels. To extract temporal information with a CNN, so-called inception blocks (Szegedy et al., 2015) are frequently used, as, for example, in Fawaz et al. (2020) and Kleinert et al. (2021). These blocks consist of individual convolutional elements with different filter sizes that are applied in parallel and are intended to learn features with different temporal localities.

RNNs offer the possibility to model nonlinear behavior in temporal sequences in a nonparametric way. Frequently used RNNs are long short-term memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) and gated recurrent unit networks (Chung et al., 2014) or hybrids of RNNs and CNNs such as in Liang and Hu (2015) and Keren and Schuller (2016). RNNs find intensive application in natural language processing, speech recognition, and signal processing, although they appear to have been largely replaced by transformer architectures more recently. However, these applications are mostly analysis problems and not predictive tasks. For time series prediction, especially for the prediction of multiple time steps into the future, there is little research evaluating the predictive performance of RNNs (Chandra et al., 2021). Moreover, Zhao et al. (2020) question the term *long* in LSTMs, as their research shows that LTSMs do not have long-term memory from a purely statistical point of view because their behavior hardly differs from that of a first-order AR. Furthermore, Cho et al. (2014) were able to show that these network types, for example, have difficulties in reflecting an annual pattern in daily-resolved data. Thus, the superposition of different periodic patterns remains a critical issue in time series prediction, as RNNs have fundamental

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

difficulties with extrapolating and predicting periodic signals (Ziyin et al., 2020) and therefore tend to focus on short-term signals only (Shih et al., 2019).

In order to deal with the superposition of different periodic signals and thus help the learning process of the NN, digital filters can be used. So-called finite impulse response (FIR) filters are realized by convolution of the time series with a window function (Oppenheim and Schafer, 1975). In fact, FIR filters are widely used in meteorology without being labeled as such, since a moving average is nothing more than a convolution with a rectangular window function. With the help of such FIR filters, it is possible to extract or remove a long-term signal from a time series or to directly divide the time series into several components with different frequency ranges, as applied, for example, in Rao and Zurbenko (1994), Wise and Comrie (2005), and Kang et al. (2013). In these studies, so-called Kolmogorov–Zurbenko (KZ) filters (Žurbenko, 1986) are used, which were specially developed for use in meteorology and promise a good separation between long-term and short-term variations of meteorological and air quality time series (Rao and Zurbenko, 1994).

There are examples of the use of filters in combination with NNs, for example, in Cui et al. (2016) and Jiang et al. (2019), but these are limited purely to analysis problems. The application of filters in a predictive setting is more complicated, because, for a prediction, filters may only be applied causally to past values, which inevitably produces a phase shift and thus a delay in the filtered signal (Oppenheim and Schafer, 1975). The lower the chosen cutoff frequency of the low-pass filter, for example, to extract the seasonal cycle, the more the resulting signal becomes delayed. This in turn leads to the fact that values in the recent past cannot be separated, as no information is yet available on the long-term components.

In this work, we propose an alternative way to filter the input time series using a composite of observations and climatological statistics to be able to separate long-term and short-term signals with the smallest possible delay. By dividing the input variables into different frequency ranges, different NN architectures are able to improve their understanding of both short-term and long-term patterns.

This paper is structured as follows: First, in Section 2, we explain and formalize the decomposition of the input time series and give details about the NN architecture used. Then, in Section 3, we describe our conducted experiments in detail, describing the data used, their preparation, the training setup, and the evaluation procedures. This is followed by the results in Section 4. Finally, we discuss our results in Section 5 and draw our conclusions in Section 6.

2. Methodology

In this paper, we combine actual observation data and a meteorologically and statistically motivated estimate of the future to overcome the issue of delay and causality (see Section 1). The estimate about the future is composed of climatological information about the seasonal as well as diurnal cycle, whereby the latter is also allowed to vary over the year. For each observation point t_0, these two time series, the observation for time steps with $t_i \leq t_0$ and the statistical estimation for $t_i > t_0$, are concatenated. By doing this, noncausal filters can be applied to the composite time series in order to separate the oscillation components of the time series such as the dominant seasonal and diurnal cycle.

The decomposition of the time series is obtained by the iterative application of several low-pass filters with different cutoff frequencies. The signal resulting from a first filter run, which only has frequencies below a given cutoff frequency, is then subtracted from the original composite signal. The next filter iteration with a higher cutoff frequency then starts on this residual, the result of which is again subtracted. By applying this cycle several times, a time series with the long-term components, multiple series covering certain frequency ranges, and a last residual time series containing all remaining short-term components are generated. Here, we test filter combinations with four and two frequency bands. The exact cycle of filtering is described in Section 2.1.

Each filtered component is finally used as an input branch of a so-called multibranch NN (MB-NN), which first processes the information of each input branch separately and then combines it in a subsequent layer. In Section 2.2, we go into more detail about the architecture of the MB-NN.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

2.1. Time series filter

For each time step t_0, a composite time series $\breve{x}_i^{(0)}$,

$$\breve{x}_i^{(0)}(t_0) = \begin{cases} x_i^{(0)}, & t_i \leq t_0, \\ a_i^{(0)}, & t_i > t_0, \end{cases} \tag{1}$$

can be created that is composed of the true observation $x_i^{(0)}$ for past time steps and a climatological estimate

$$a_i^{(0)} = \bar{x}_{\text{month}}^{(0)}(t_i) + \Delta_{\text{hour}}^{(0)}(t_i) \tag{2}$$

for future values. The composite time series $\breve{x}_i^{(0)}$ is always a function of the current observation time t_0. The climatological estimate is derived from a monthly mean value $\bar{x}_{\text{month}}^{(0)}(t_i)$ with

$$\bar{x}_{\text{month}}^{(0)}(t_i) = f^{(0)}\left(x_i^{(0)}\right) \tag{3}$$

and a daily anomaly $\Delta_{\text{hour}}^{(0)}(t_i)$ of it with

$$\Delta_{\text{hour}}^{(0)}(t_i) = g^{(0)}\left(x_i^{(0)} - \bar{x}_{\text{month}}^{(0)}(t_i)\right) \tag{4}$$

that may vary over the year. $f^{(0)}$ and $g^{(0)}$ are arbitrary functions used to calculate these estimates. The composite time series $\breve{x}_i^{(0)}(t_0)$ can then be convolved with an FIR filter with given properties $b_i^{(0)}$. The result of this convolution is a low-pass filtered time series:

$$\tilde{x}_n^{(0)}(t_0) = \sum_{i=t_0-N/2}^{t_0+N/2} b_i^{(0)} \cdot \breve{x}_{n-i}^{(0)}(t_0). \tag{5}$$

It should be noted again that $\tilde{x}_i^{(0)}$ is still a function of the current observation time t_0. From the composite time series and its filtered result, a residual

$$x_i^{(1)}(t_0) = x_i^{(0)} - \tilde{x}_i^{(0)}(t_0) \tag{6}$$

can be calculated, which represents the equivalent high-pass signal.

A new filtering step can now be applied to the residual $x_i^{(1)}(t_0)$. For this, the a priori information, which is used to estimate the future, is first newly calculated. Ideally, if the first filter application in equation (5) has already completely removed the seasonal cycle, the climatological mean $\bar{x}_{\text{month}}^{(1)}(t_i)$ is zero, and based on our assumption in equation (2), only an estimate of the hourly daily anomaly $\Delta_{\text{hour}}^{(1)}(t_i)$ remains. With this information, a composite time series $\breve{x}_i^{(1)}(t_0)$ can now be formed, which can separate higher frequency oscillation components using another low-pass filter with a higher cutoff frequency. A time series $\tilde{x}_n^{(1)}(t_0)$ created in this way corresponds to the application of a band-pass filter. On the residual $x_i^{(2)}(t_0)$, the next filter iteration with corresponding a priori information can be carried out. Generalized, equations (1)–(6) result in

$$\bar{x}_{\text{month}}^{(j)}(t_i) = f^{(j)}\left(x_i^{(j)}\right), \tag{7}$$

$$\Delta_{\text{hour}}^{(j)}(t_i) = g^{(j)}\left(x_i^{(j)} - \bar{x}_{\text{month}}^{(j)}(t_i)\right), \tag{8}$$

$$a_i^{(j)} = \bar{x}_{\text{month}}^{(j)}(t_i) + \Delta_{\text{hour}}^{(j)}(t_i), \tag{9}$$

$$\breve{x}_i^{(j)}(t_0) = \begin{cases} x_i^{(j)}, & t_i \leq t_0, \\ a_i^{(j)}, & t_i > t_0, \end{cases} \tag{10}$$

$$\tilde{x}_n(j)(t_0) = \sum_{i=t_0-N/2}^{t_0+N/2} b_i^{(j)} \cdot \breve{x}_{n-i}^{(j)}(t_0), \tag{11}$$

$$x_i^{(j+1)}(t_0) = x_i^{(j)} - \tilde{x}_i^{(j)}(t_0). \tag{12}$$

If a time series was decomposed according to this procedure using equations (7)–(12) with J filters, it now consists of a component $\tilde{x}^{(0)}$, that contains all low-frequency components, $J-1$ components $\tilde{x}^{(j)}$ with oscillations on different frequency intervals, and a residual term $x^{(J)}$ that only covers the high-frequency components. The original signal can be completely reconstructed at any time t_i by summing up the individual components.

In this study, oscillation patterns that have a periodicity of months or years are separated from the series in the first filter iteration by using a cutoff period of 21 days, which is motivated by the work of Kang et al. (2013). We also consider a cutoff period of around 75 days, as used, for example, in Rao et al. (1997) and Wise and Comrie (2005), and evaluate the impact of this low-frequency cutoff. For the further decomposition of the time series, we first follow the cutoff frequencies proposed in Kang et al. (2013) and divide the time series into the four components baseline (BL, period >21 days), synoptic (SY, period >2.7 days), diurnal (DU, period >11 hr), and intraday (ID, residuum). Since Kang et al. (2013) found that a clear separation of the individual components is not possible for the short-term components, but can be achieved between the long-term and short-term components, we conduct a second series of experiments in which the input data are only divided into long term (LT, period >21 days) and short term (ST, residuum).

Figure 1 shows the result of such a decomposition into four components. It can be seen that the BL component decreases with time. The SY component fluctuates around zero with a moderate oscillation between August 16 and 20. In the DU component, the day-to-day variability and diurnal oscillation patterns are visible, and in the ID series, several positive and negative peaks are apparent. Overall, it can be seen that the climatological statistical estimation of the future provides a reliable prediction. However, since a slightly higher ozone episode from August 25 onward cannot be covered by the climatology, the long-term component BL is slightly underestimated, but for time points up to t_0, this has hardly any effect. This small difference of a few parts per billion (ppb) is covered by the SY and DU components, so that the residual component ID no longer contains any deviations.

2.2. Multibranch NN

The time series divided into individual components according to Section 2.1 serves as the input of an MB-NN. In this work, we investigate three different types of MB-NNs based on fully connected, convolutional, or recurrent layers. We therefore refer to the corresponding NNs in the following as MB-FCN, MB-CNN, and MB-RNN. The respective filter components of all input variables are presented together to one branch each. Thereby, each filter component leads to a distinct input branch in the NN. A branch first learns the local characteristics of the oscillation patterns and can therefore also be understood as its own subnetwork. Afterward, the MB-NN can learn global links, that is, the interaction of the different scales, by a learned (nonlinear) combination of the individual branches in a subsequent network. However, the individual branches are not trained separately, but the error signal propagates from the very last layer backward through the entire network and then splits up between the individual branches.

The sample MB-NN shown on the left in Figure 2 consists of four input branches, each receiving a component from the long-term $\tilde{x}^{(0)}$ (BL) to the residual $x^{(3)}$ (ID) of the filter decomposition. Here, the data presented as example input are the same as in Figure 1, but each component has already been scaled to a mean of zero and a standard deviation of 1, taking into account several years of data. In addition to the characteristics of the example already discussed in Section 2.1, it can be seen from the scaling that the BL component is above the mean, indicating a slightly increased long-term ozone concentration. The SY component, on the other hand, shows only a weak fluctuation. The data are fed into four different branches, each of which consists of an arbitrary architecture based on fully connected, convolutional, or

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

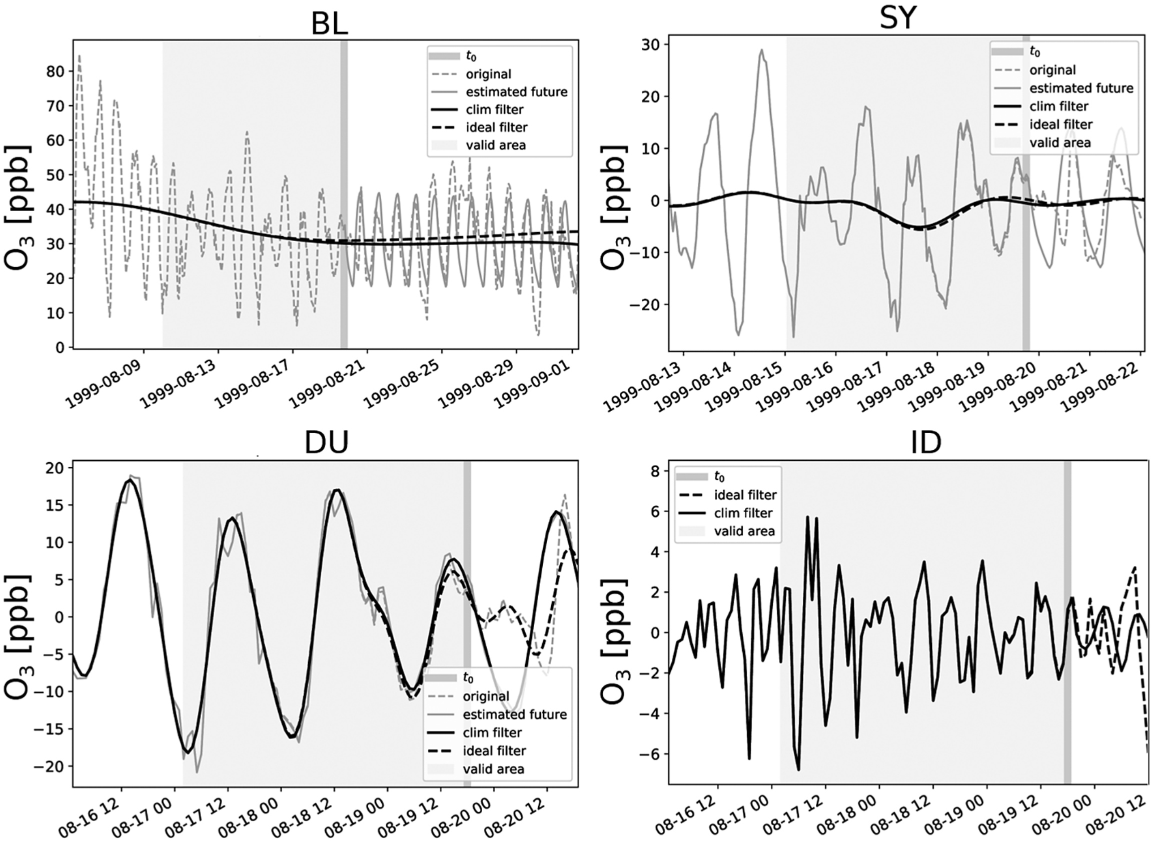


Figure 1. Decomposition of an ozone time series into baseline (BL), synoptic (SY), diurnal (DU), and intraday (ID) components at $t_0 =$ August 19, 1999 (dark gray background) at an arbitrary sample site (here DEMV004). Shown are the true observations $x_i^{(j)}$ (dashed light gray), the a priori estimation $a_i^{(j)}$ about the future (solid light gray), the filtering of the time series composed of observation and a priori information $x_i(j)(t_0)$ (solid black), and the response of a noncausal filter with access to future values (dashed black) as a reference for a perfect filtering. Because of boundary effects, only values inside the marked area (light gray background) are valid.

recurrent layers. Subsequently, the information of these four subnetworks is concatenated and parsed in the tail to a concluding neural block, which finally results in the output layer.

On the right side in Figure 2, only the decomposition into the two components LT and ST is applied. Since the cutoff frequency is the same for LT and BL, the LT input is equal to the BL input. All short-term components are combined and fed to the NN in the form of the ST component. This arbitrary MB-NN again uses a specified type of neural layers in each branch before the information is interconnected in the concatenate layer and then processed in the subsequent neural block, which finally leads to the output again. Figure 2 shows a generic view of the four-branch and two-branch NNs. The specific architectures employed in this study are depicted in Section 3.3, Tables B2 and B3 in Appendix B, and Figures D1–D5 in Appendix D.

3. Experiment Setup

For data preprocessing and model training and evaluation, we employ the software MLAir (version 2.0.0; Leufen et al., 2022). MLAir is a tool written in Python that was developed especially for the application of ML to meteorological time series. The program executes a complete life cycle of an ML training, from preprocessing to training and evaluation. A detailed description of MLAir can be found in Leufen et al. (2021).

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

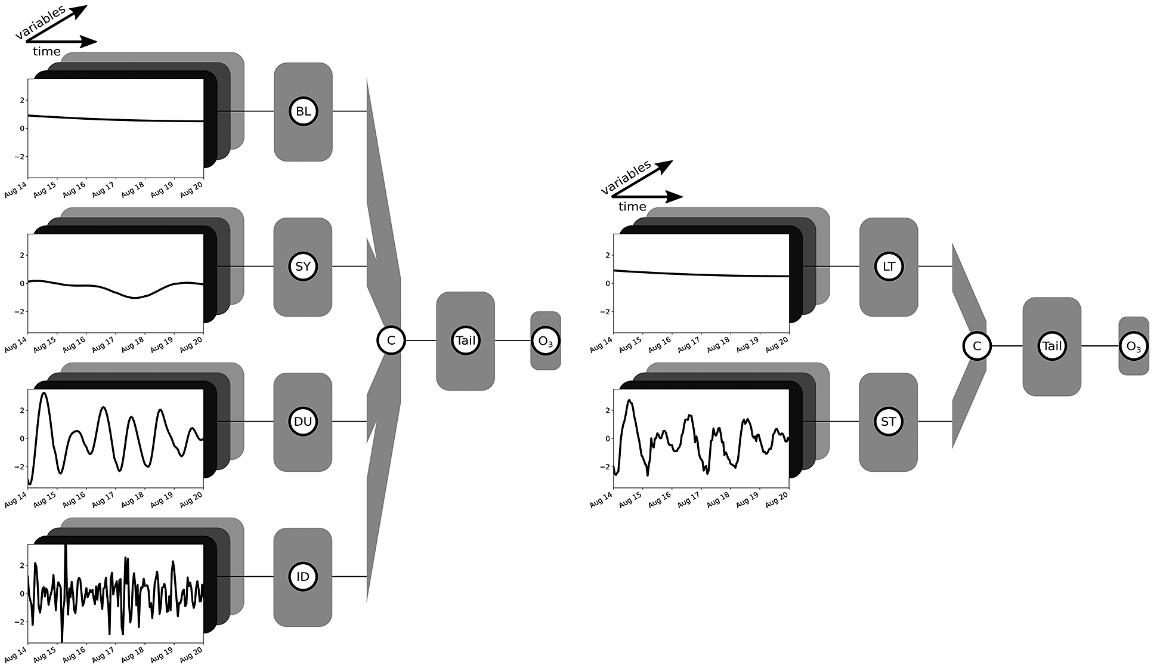


Figure 2. *Sketching of two arbitrary MB-NNs with inputs divided into four components (BL, SY, DU, and ID) on the left and two components (LTand ST) on the right. The input example shown here corresponds to the data shown in Figure 1, whereby the components SY, DU, and ID on the right-hand side have not been decomposed, but rather grouped together as the short-term component ST. Moreover, the data have already been scaled. Each input component of a branch consists of several variables, indicated schematically by the boxes in different shades of gray. The boxes identified by the branch name, also in gray, each represent an independent neuronal block with user-defined layer types such as fully connected, convolutional, or recurrent layers and any number of layers. Subsequently, the branches are then combined via a concatenation layer marked as "C." This is followed by a final neural block labeled as "Tail," which can also have any configuration and finally ends in the output layer of the NN indicated by the tag "O_3." The sketches are based on a visualization with the Net2Vis tool (Bauerle et al., 2021).*

3.1. Data

In this study, data from the Tropospheric Ozone Assessment Report database (TOAR DB; Schultz et al., 2017) are used. This database collects global in situ observation data with a special focus on air quality and in particular on ozone. As part of the Tropospheric Ozone Assessment Report (TOAR, 2021), over 10,000 air quality measuring stations worldwide were inserted into the database. For the area over Central Europe, these observations are supplemented by model reanalysis data interpolated to the measuring stations, which originate from the consortium for small-scale modeling (COSMO) reanalysis with 6-km horizontal resolution (COSMO-REA6; Bollmeyer et al., 2015). The measured data provided by the German Environment Agency (Umweltbundesamt) are available in hourly resolution.

By following Kleinert et al. (2021), we choose a set of nine input variables. As regards chemistry, we use the observation of O_3 as well as the measured values of NO and NO_2, which are important precursors for ozone formation. In this context, it would be desirable to include other chemical variables and especially volatile organic compounds (VOCs), such as isoprene and acetaldehyde, which have a crucial influence on the ozone production regime (Kumar and Sinha, 2021). However, the measurement coverage of VOCs is very low, so that only very sporadic recordings are available, which would result in a rather small dataset. Concerning meteorology, in addition to the wind in its individual components as well as the height of the planetary boundary layer as an indicator for advection and mixing, we use temperature and the cloud cover as a proxy for solar irradiance, and the relative humidity. All meteorological variables are

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Table 1. Input and target variables with respective temporal resolution and origin. Data labeled with UBA originate from measurement sites provided by the German Environment Agency, and data with flag COSMO-REA6 have been taken from reanalysis.

	Variable	Origin	Temporal resolution
Input	NO	UBA	1 hr
	NO_2	UBA	1 hr
	O_3	UBA	1 hr
	Cloud cover	COSMO-REA6	1 hr
	Planetary boundary layer height	COSMO-REA6	1 hr
	Relative humidity	COSMO-REA6	1 hr
	Temperature	COSMO-REA6	1 hr
	Wind's u-component	COSMO-REA6	1 hr
	Wind's v-component	COSMO-REA6	1 hr
Target	dma8eu O_3	UBA	1 day

Abbreviations: COSMO-REA6, consortium for small-scale modeling reanalysis with 6-km horizontal resolution; UBA, Umweltbundesamt.

extracted from COSMO-REA6, but are treated in the following as if they were observations at the measuring stations. Table 1 provides an overview of the observation and model variables used. The target variable ozone is also obtained directly from the TOAR DB. Rather than using the hourly values, however, the daily aggregation to the daily maximum 8-hr average value according to the European Union definition (dma8eu) is performed by the TOAR DB and extracted directly in daily resolution. It is important to note that the calculation of dma8eu includes observations from 5 p.m. of the previous day (cf. European Parliament and Council of the European Union, 2008). Care must therefore be taken that ozone values from 5 p.m. on the day of t_0 may no longer be used as inputs to ensure a clear separation, as they are already included in the calculation of the target value.

This study is based on a relatively homogeneous dataset, so that the NNs can learn better and thus the effect due to time series filtering becomes clearer. In order to obtain such a dataset of observations, we restrict our investigations to the area of the North German Plain, which includes all areas in Germany north of 52.5°N. We choose this area because of the rather flat terrain; no station is located higher than 150 m above sea level. In addition, we restrict ourselves to measurement stations that are classified as background according to the European Environmental Agency AirBase classification (European Parliament and Council of the European Union, 2008), which means that no industry or major road is located in the direct proximity of the stations and consequently the pollution level of this station is not dominated by a single source. All stations are located in a rural or suburban environment. These restrictions result in a total number of 55 stations distributed over the entire area of the North German Plain. A geographical overview can be found in Figure A2 in Appendix A. It should be noted that no measuring station provides complete time series, so that gaps within the data occur. However, since the filter approach requires continuous data, gaps of up to 24 consecutive hours on the input side and gaps of 2 days on the target side are filled by linear interpolation along time.

3.2. Preparing of input and target data

The entire dataset is split along the temporal axis into training, validation, and test data. For this purpose, all data in the period from January 1, 1997 to December 31, 2007 are used for training. The a priori information of the time series filter about seasonal and diurnal cycles is calculated based on this set. The following 2 years, January 1, 2008 to December 31, 2009, are used for the validation of the training, and

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Table 2. Number of measurement stations and resulting number of samples used in this study. All stations are classified as background and situated either in a rural or suburban surrounding in the area of the North German Plain. Data are split along the temporal axes into three subsequent blocks for training, validation, and testing.

	Training	Validation	Testing
Stations			
Rural	31	17	17
Suburban	24	15	13
Total	55	32	30
Samples			
Rural	54,544	10,927	16,858
Suburban	40,968	10,405	13,622
Total	95,512	21,332	30,480

all data from January 1, 2010 onward are used for the final evaluation and testing of the trained model. For the meteorological data, there are no updates in the TOAR DB since January 1, 2015, so more recent air quality measurements cannot be used in this study.

For each time step t_0, the time series is decomposed using the filter approach as defined in Section 2.1. The a priori information is obtained from the training dataset alone so that validation and test datasets remain truly independent. Afterward, the input variables are standardized so that each filter component of each variable has a mean of zero and a standard deviation of 1 (Z-score normalization). For the target variable dma8eu ozone, we choose the Z-score normalization as well. All transformation properties for both inputs and targets are calculated exclusively on the training data and applied to the remaining subsets. Moreover, these properties are not determined individually per station, but jointly across all measuring stations.

In this work, we choose the number of past time steps for the input data as 65 hr. This corresponds to the three preceding days minus the measurements starting at 5 p.m. on the current day of t_0 due to the calculation procedure of dma8eu as already mentioned. The number of time steps to be predicted is set to the next 4 days for the target. All in all, we use almost 100,000 training samples and 20,000 and 30,000 samples for validation and testing, respectively (see Table 2 for exact numbers). The data availability at individual stations, as well as the total number of different stations at each point in time, is shown in Figures A1 and A3 in Appendix A, respectively. The visible larger data gaps are caused by a series of missing values that exceed the maximum interpolation length.

3.3. Training setup and hyperparameter search

First, we search for an optimal decomposition of the input time series for the NNs by optimizing the hyperparameters for the MB-FCN. Second, we use the most suitable decomposition and train different MB-CNNs and MB-RNNs on these data. Finally, we train equivalent network architectures without decomposition of the input time series to obtain a direct comparison of the decomposition approach as outlined in Section 3.4. All experiments are assessed based on the mean square error (MSE), as presented in Section 3.5. Since we are testing a variety of different models, we have summarized the most relevant abbreviations in Table 3.

The experiments to find an optimal decomposition of the inputs and best hyperparameters for the MB-FCN start with the same cutoff frequencies for decomposition as used in Kang et al. (2013), who divide their data into the four components BL, SY, DU, and ID, as explained in Section 2.1. Since there is generally no optimal a priori choice for a filter (Oppenheim and Schafer, 1975) and furthermore this is

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Table 3. Summary of model acronyms used in this study depending on their architecture and the number of input branches. The abbreviations for the branch types refer to the unfiltered original raw data and either to the temporal decomposition into the four components baseline (BL, period >21 days), synoptic (SY, period >2.7 days), diurnal (DU, period >11 hr), and intraday (ID, residuum), or to the decomposition into two components long term (LT, period >21 days) and short term (ST, residuum). When multiple input components are used, as indicated in the column labeled Count, the NNs are constructed with multiple input branches, each receiving a single component, and are therefore referred to as multibranch (MB). For technical reasons, this MB approach is not applicable to the OLS model, which instead uses a flattened version of the decomposed inputs and is therefore not specified as MB.

Input branches		Model name			
Branch type(s)	Count	FCN	CNN	RNN	OLS
Raw	1	FCN	CNN	RNN	OLS
LT and ST	2	MB-FCN-LT/ST	MB-CNN-LT/ST	MB-RNN-LT/ST	OLS-LT/ST
LT, ST, and raw	3	MB-FCN-LT/ST+raw	–	–	–
BL, SY, DU, and ID	4	MB-FCN-BL/SY/DU/ID	–	–	–
BL, SY, DU, ID, and raw	5	MB-FCN-BL/SY/DU/ID+raw	–	–	–

Abbreviations: CNN, convolutional neural network; FCN; fully connected network; OLS, least squares regression.

likely to vary from one application to another, we choose a Kaiser filter (Kaiser, 1966) with a beta parameter of $\beta = 5$ for the decomposition of the time series. We prefer this filter for practical considerations, as a filter with a Kaiser window features a sharper gain reduction in the transition area at the cutoff frequency in comparison to the KZ filter. Based on this, we test a large number of combinations of the hyperparameters (see Table B1 in Appendix B for details). The trained MB-FCN with the lowest MSE on the validation in this experiment is referred to as MB-FCN-BL/SY/DU/ID in the following. Since, as already mentioned in Section 2.1, a clear decomposition in individual components is not always possible, we start a second series of experiments in which the input data are only divided into long term (LT) and short term (ST). We tested cutoff periods of 75 (Rao et al., 1997; Wise and Comrie, 2005) and 21 days (Kang et al., 2013), and found no difference with respect to the MSE of the trained networks. Hence, we selected the cutoff period of 21 days and refer to the trained network as MB-FCN-LT/ST in the following. After finding an optimal set of hyperparameters for both experiments, we vary the input data and study the resulting effect on the prediction skill. In two extra experiments, we add an additional branch with the unfiltered raw data to the inputs. According to the previous labels, these experiments result in the NNs labeled MB-FCN-BL/SY/DU/ID+raw and MB-FCN-LT/ST+raw. We have summarized the optimal hyperparameters for each of the MB-FCN architectures in Table B2 in Appendix B.

Based on the findings with the MB-FCNs, we choose the best MB-FCN and the corresponding preprocessing and temporal decomposition of the input time series for the second part of the experiments, in which we test more sophisticated network architectures. With the data remaining the same, we investigate to what extent using MB-CNN or MB-RNN leads to an improvement compared to MB-FCN and also in relation to their counterparts without temporal decomposition (CNN and RNN). For this purpose, we test different architectures for CNN and RNN with and without temporal decomposition separately and compare the best representative found by the experiment, respectively. The optimal hyperparameters given by this experiments are outlined in Table B3 in Appendix B, and a visualization of the best NNs can be found in Figures D1–D5 in Appendix D. Regarding the CNN architecture, we varied the total number of layers and filters in each layer, the filter size, the use of pooling layers, as well as the application of convolutional blocks after the concatenate layer and the layout of the final dense layers. For the RNNs, during hyperparameter search, we used different numbers of LSTM cells per layer and tried stacked LSTM layers. Furthermore, we added recurrent layers after the concatenate layer in some experiments. In general, we tested different dropout rates, learning rates, a decay of the learning rate, and several activation functions.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

3.4. Reference forecasts

We compare the results of the trained NNs with a persistence forecast, which generally performs well on short-term predictions (Murphy, 1992; Wilks, 2006). The persistence consists of the last observation, in this case the value of dma8eu ozone on the day of t_0, which serves as a prediction for all future days. We also compare the results with climatological reference forecasts following Murphy (1988). Details are given in Section 3.5. Furthermore, we compare the MB-NNs to an ordinary least squares regression (OLS), an FCN, a CNN, and an RNN. The basis for these competitors is hourly data without special preparation, that is, without prior decomposition into the individual components. The parameters of the OLS are created analogously to the NNs on the training data only. For the FCN, CNN, and RNN, sets of optimal parameters were determined experimentally in preliminary experiments also on training data. Only the NNs with the lowest MSE on the validation data are shown here. Furthermore, as with MB-NNs, we apply an OLS method to the temporally decomposed input data. For technical reasons, the OLS approach is not able to work with branched data and therefore uses flattened inputs instead. Finally, we draw a comparison with the IntelliO3-ts model from Kleinert et al. (2021). IntelliO3-ts is a CNN based on inception blocks (see Section 1). In contrast to the study here, IntelliO3-ts was trained for the entire area of Germany. It should be noted that IntelliO3-ts is based on daily aggregated input data, whereas all NNs trained in this study use an hourly resolution of input data. For all models, the temporal resolution of the targets is daily, so that the NNs of this study have to deal with different temporal resolutions, which does not apply for IntelliO3-ts.

3.5. Evaluation methods

The evaluation of the NNs takes place exclusively on the test data that are unknown to the models. To assess the performance of the NNs, we examine both absolute and relative measures of accuracy. Accuracy measures generally represent the relationship between a prediction and the value to be predicted. Typically, for an absolute measure of the predictive quality on continuous values, the MSE is used. The MSE is a good choice as a measure because it takes into account the bias as well as the variances and the correlation between prediction and observation. To determine the uncertainty of the MSE, we choose a resampling test procedure (cf. Wilks, 2006). Due to the large amount of data, a bootstrap approach is suitable. Synthetic datasets are generated from the test data by repeated blockwise resampling with replacement. For each set, the error, in our case the MSE, is calculated. With a sufficiently large number of repetitions (here $n = 1,000$), we can access an estimate of the error uncertainty. To reduce misleading effects caused by autocorrelation, we divide the test data along the time axis into monthly blocks and draw from these instead of the individual values.

To compare individual models directly with each other, we derive a skill score from the MSE as a relative measure of accuracy. In this study, the skill score always consists of the MSE of the actual forecast as well as the MSE of the reference forecast and is given by

$$\mathrm{SS} = 1 - \frac{\mathrm{MSE}}{\mathrm{MSE_{ref}}}. \tag{13}$$

Accordingly, a value around zero means that no improvement over a reference could be achieved. If the skill score is positive, an improvement can generally be assumed, and if it is negative, the prediction accuracy is below the reference.

For the climatological analysis of the NN, we refer to Murphy (1988), who determines the climatological quality of a model by breaking down the information into four cases. In Case 1, the forecast is compared with an annual mean calculated on data that are known to the model. For this study, we consider both the training and validation data to be internal data, since the NN used these data during training and hyperparameter search. Case 2 extends a climatological consideration by differentiating into 12 individual monthly averages. Cases 3 and 4, respectively, are the corresponding transfers of the aforementioned analyses, but on test data that are unknown to the model.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Another helpful method for the verification of predictions is the consideration of the joint distribution $p(y_i, o_j)$ of prediction y_i and observation o_j (Murphy and Winkler, 1987). The joint distribution can be factorized to shed light on particular facets. With the calibration-refinement factorization

$$p(y_i, o_j) = p(o_j | y_i) \cdot p(y_i), \tag{14}$$

the conditional probability $p(o_j|y_i)$ and the marginal distribution of the prediction $p(y_i)$ are considered. $p(o_j|y_i)$ provides information on the probability of each possible event o_j occurring when a value y_i is predicted, and thus how well the forecast is calibrated. $p(y_i)$ indicates the relative frequency of the predicted value. It is desirable to have a distribution of y with a width equal to that for o.

3.6. Feature importance analysis

Due to the NN's nonlinearity, the influence of individual inputs or variables on the model is not always directly obvious. Therefore, we again use a bootstrap approach to gain insight into the feature importance. In general, we remove a certain piece of information and examine the skill score in comparison to the original prediction to see whether the prediction quality of the NN decreases or increases as a result. If the skill score remains constant, this is an indication that the examined information does not provide any additional benefit for the NN. The more negative the skill score becomes in the feature importance analysis, the more likely it is that the examined variable contains important information for the NN. In the unlikely case of a positive skill score, it can be inferred that the context of this variable was learned incorrectly and thus disturbs the prediction.

For the feature importance analysis, we take a look at three different cases. First, we analyze the influence of the temporal decomposition by destroying the information of an entire input branch, for example, all low-frequent components (BL resp. LT). This yields information about the effect of the different time scales from long term to short term and the residuum. In the second step, we adopt a different perspective and look at complete variables with all temporal components (e.g., both LT and ST components of temperature). In the third step, we drive down one tier and consider each input separately to get information about whether a single input has a very strong influence on the prediction (e.g., BL component of NO_2).

To break down the information for the feature importance analysis, we randomly draw the quantity to be examined from its observations. Statistically, a test variable obtained in this way is sampled from the same distribution as the original variable. However, the test variable is detached from its temporal context as well as from the context of other variables. This procedure is repeated 100 times to reduce random effects.

The feature importance analysis considers only the influence of a single quantity and no pairwise or further correlations. However, the isolated approach already provides relevant information about the feature importance. It is important to note that this analysis can only show the importance of the inputs for the trained NN and that no physical or causal relationships can be deduced from this kind of analysis in general.

4. Results

Since a comparison of all models against each other would quickly become incomprehensible, we first look at the results of the resampling in order to obtain a ranking of MB-FCNs (see Table 3 for a summary of model acronyms). The results of the bootstrapping are shown in Figure 3 and listed also in Table C1 in Appendix C. With a block length of 1 month and 1,000 repetitions of the bootstrapping, it can be seen that the simple FCN cannot adequately represent the relationships between inputs and targets in comparison to the other models. Moreover, it is visible that the performance of the MB-FCN-BL/SY/DU/ID falls behind in comparison to the other MB-FCNs with an average MSE $> 70\,\mathrm{ppb}^2$. The smallest resampling errors could be achieved with the models MB-FCN-BL/SY/DU/ID+raw, MB-FCN-LT/ST, and MB-FCN-LT/ST+raw.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

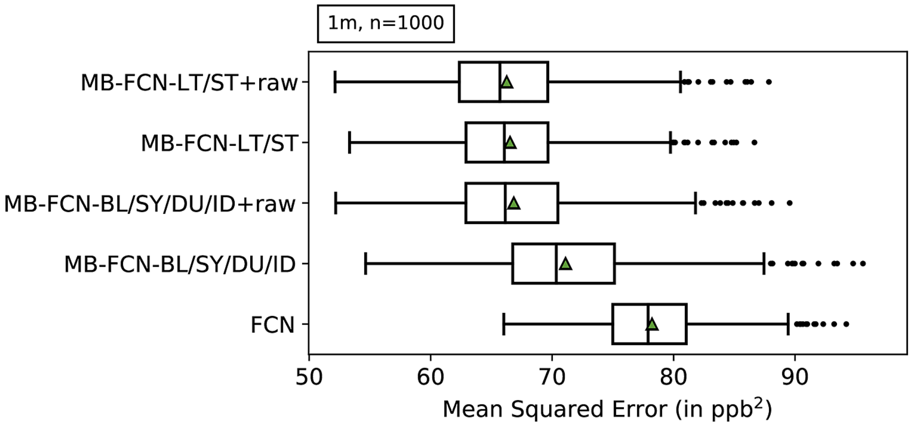


Figure 3. *Results of the uncertainty estimation of the MSE using a bootstrap approach represented as box-and-whiskers. For each model, the median is shown as a black vertical line, the mean as a green triangle, the upper and lower quartiles in the form of the box, the upper and lower whiskers, which correspond to 1.5 times the interquartile range, and outliers beyond the whiskers as individual data points. The models are ordered from top to bottom with ascending average MSE. A total of 1,000 bootstrap samples were created by resampling with the replacement of single-month blocks.*

When comparing the decomposition into BL/SY/DU/ID and the decomposition into LT/ST components, the latter decomposition tends to yield a lower error. Alternatively, it is possible to achieve comparative performance by adding the raw data to both variants of decomposition. For the LT/ST decomposition, however, this improvement is marginal.

Since the forecast accuracy of the top three NNs is nearly indistinguishable, especially for the two models with the LT/ST split, we choose the MB-FCN-LT/ST network and so the LT/ST decomposition for further analysis, since, of the three winning candidates, this is the network with the smallest number of trainable parameters (see Table B2 in Appendix B).

So far, we have shown the advantages of an LT/ST decomposition during preprocessing for FCNs. Therefore, in the following, we apply our proposed decomposition to more elaborated network architectures, namely a CNN and an RNN architecture. We again consider the uncertainty estimation of the MSE using the bootstrap approach and calculate the skill score with respect to the MSE in pairs for an NN type that was trained once as an MB-NN with temporally decomposed inputs and once with the raw hourly data. Similarly, we consider the skill score of OLS on decomposed and raw data, respectively. The results are shown in Figure 4. It can be seen that the skill score is always positive for all models. This in turn means that using our proposed time decomposition of the input time series improves all the models analyzed here. When looking at the individual models, it can be differentiated that the FCN architecture in particular benefits from the decomposition, whereas the improvement is smaller for RNN and smallest but still significant for OLS and CNN.

Based on the uncertainty estimation of the MSE shown in Figure 5 and also listed in Table C2 in Appendix C, the models can be roughly divided into three groups according to their average MSE. The last group consists solely of the persistence prediction, which delivers a significantly worse prediction than all other methods and lies at an MSE of 107 ppb^2 on average. In the intermediate group with an MSE between 70 and 80 ppb^2, only approaches that do not use temporally decomposed inputs are found, including the IntelliO3-ts-v1 model. Overall, the FCN performs worst with a mean MSE of 78 ppb^2, and the best results in this group are achieved with the CNN. In the leading group are exclusively methods that rely on the decomposition of the input time series. The OLS with the LT/ST decomposition has the highest error within this group with 68 ppb^2. The lowest errors can be obtained with the MB-FCN and the MB-RNN, whereby the MSE for both NNs is around 66 ppb^2.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

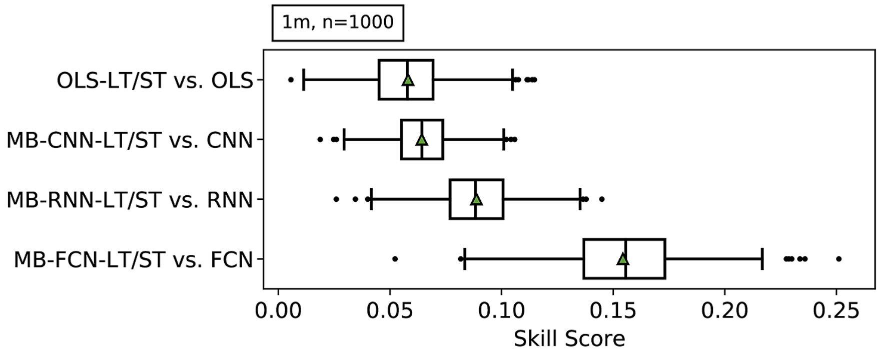


Figure 4. *Pairwise comparison of different models running with temporal decomposed or raw data by calculating the skill score on the results from the uncertainty estimation of the mean square error using a bootstrap approach represented as box-and-whiskers. For each model, the median is shown as a black vertical line, the mean as a green triangle, the upper and lower quartiles in the form of the box, the upper and lower whiskers, which correspond to 1.5 times the interquartile range, and outliers beyond the whiskers as individual data points. A total of 1,000 bootstrap samples were created by resampling with the replacement of single-month blocks.*

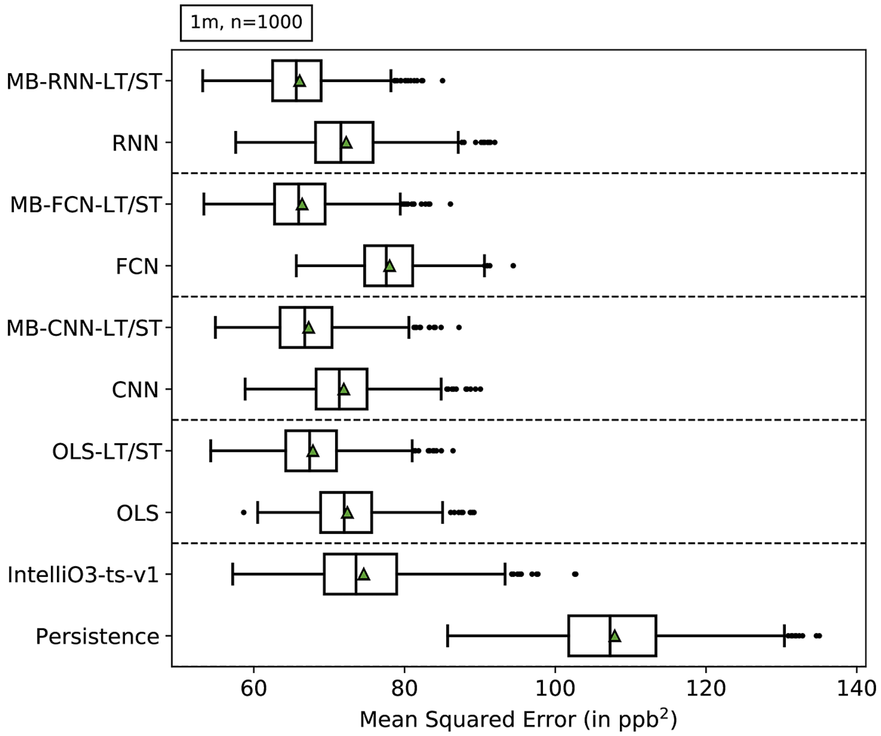


Figure 5. *The same as Figure 3, but for a different set of models. Results of the uncertainty estimation of the MSE using a bootstrap approach represented as box-and-whiskers. For each model, the median is shown as a black vertical line, the mean as a green triangle, the upper and lower quartiles in the form of the box, the upper and lower whiskers, which correspond to 1.5 times the interquartile range, and outliers beyond the whiskers as individual data points. The models are ordered from top to bottom with ascending average MSE. A total of 1,000 bootstrap samples were created by resampling with the replacement of single-month blocks. Note that the uncertainty estimation shown here is independent of the results shown in Figure 3, and therefore numbers may vary for statistical reasons.*

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

In order to understand why the decomposition consistently brings about an improvement for all methods considered here, we look exemplarily at the MB-FCN-LT/ST in more detail in the following. However, it should be mentioned that the discussed aspects are also basically valid for the other NN types.

First, we have a look at the calibration-refinement factorization of the joint distribution (Figure 6) according to equation (14). It can be seen that the distribution of the forecasted concentration of ozone becomes narrower toward the mean with increasing lead time. While the MB-FCN-LT/ST is still able to predict values of >70 ppb for the 1-day forecast, it is limited to values below 60 ppb for the 4-day forecast and tends to underestimate larger concentrations with increasing lead time. According to the conditional probability of observing an issued forecast, the MB-FCN-LT/ST is best calibrated for the first forecast day and especially in the value range from 20 to 60 ppb. However, observations of high ozone concentrations, starting from values above 60 ppb, are generally underestimated by the NN. Coupled with the already mentioned narrowing of the forecast's distribution, the underestimation of high ozone concentrations increases with lead time.

The shortcomings with the prediction of the tails of the distribution of observations are also evident when looking at the seasonal behavior of the MB-FCN-LT/ST. Figure 7 summarizes the distribution of observations and predictions of the NN for each month. The narrowing toward the mean with increasing lead time is also clearly visible here in the whiskers and the interquartile range in the form of the box. However, it can already be observed that, from a climatological perspective, the forecasts are in the range

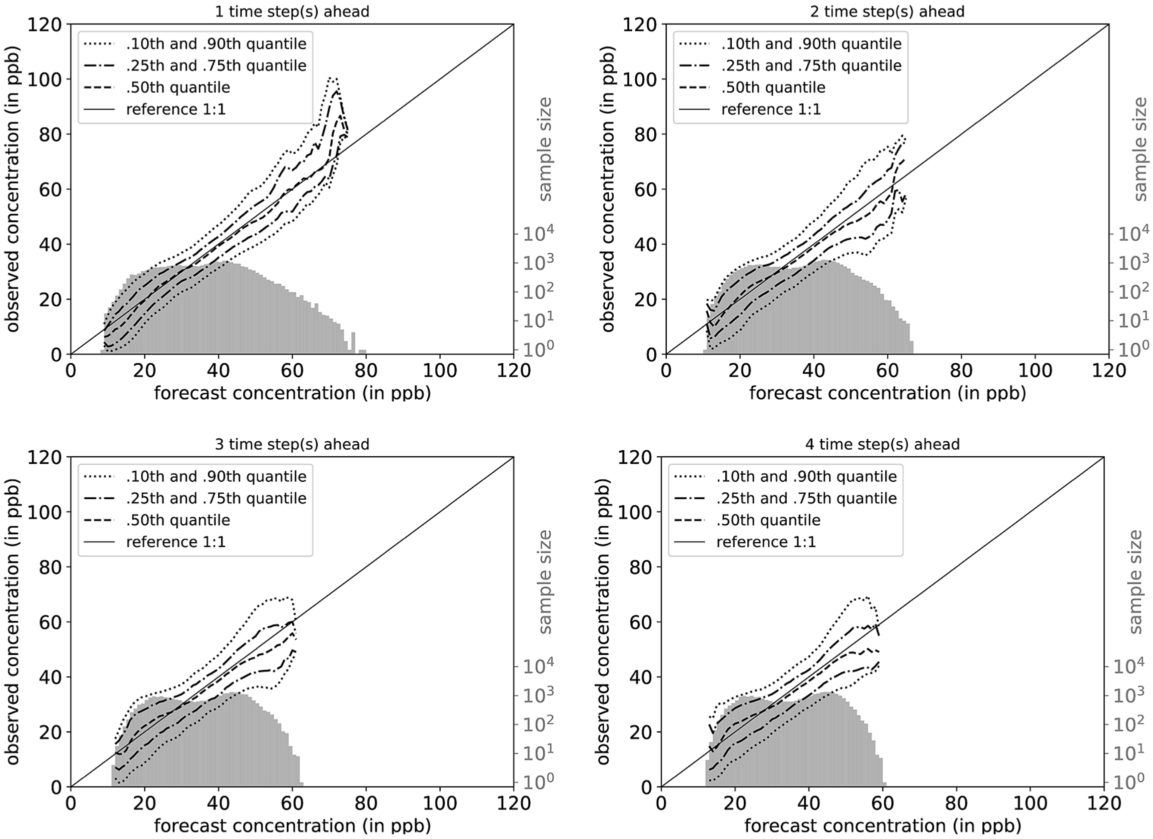


Figure 6. Joint distribution of prediction and observation in the calibration-refinement factorization $p(y_i, o_j)$ for the MB-FCN-LT/ST for all four lead times. On the one hand, the marginal distribution $p(y_i)$ of the prediction is shown as a histogram in gray colors with the axis on the right, and on the other hand, the conditional probability $p(o_j|y_i)$ is expressed by quantiles in the form of differently dashed lines. The reference line of a perfectly calibrated forecast is also shown as a solid line.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

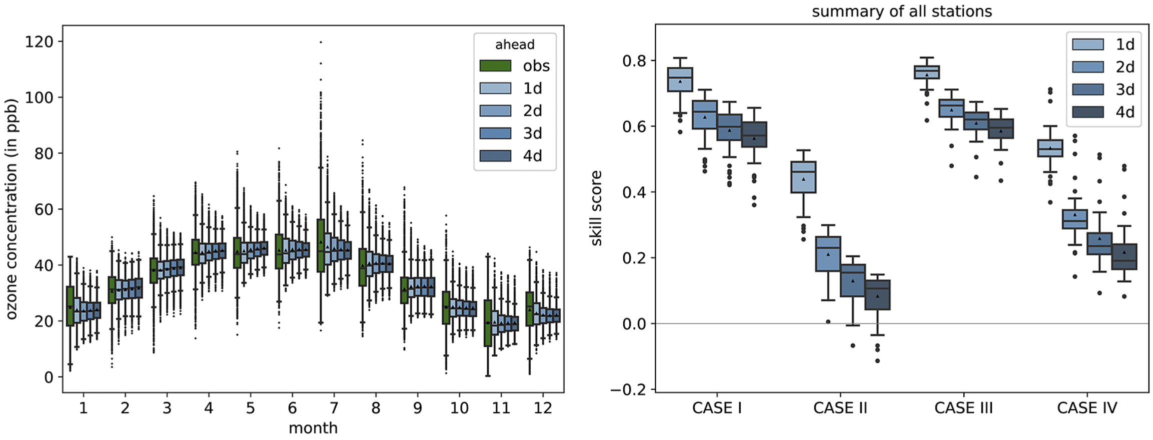


Figure 7. *Overview of the climatological behavior of the MB-FCN-LT/ST forecast shown as a monthly distribution of the observation and forecasts on the left and the analysis of the climatological skill score according to Murphy (1988) differentiated into four cases on the right. The observations are highlighted in green and the forecasts in blue. As in Figure 3, the data are presented as box-and-whiskers, with the black triangle representing the mean.*

of the observations, and the annual cycle of the ozone concentration can be modeled. Yet the month of July stands out in particular, where it is clearly recognizable that the NN is not able to represent the large variability of values from 20 ppb to values over 100 ppb that occur during summer.

The direct comparison according to Murphy (1988) between the climatological annual mean of the observation and the forecast of the NN for the training and validation data (Case 1) as well as for the test data (Case 3) shows a high skill score in favor of the NN compared to the single-valued climatological reference as the NN captures the seasonal cycle (Figure 7). Furthermore, in direct comparison with the climatological monthly means (Cases 2 and 4), the MB-FCN-LT/ST can achieve an added value in terms of information. However, the skill score on all datasets decreases gradually with longer lead times. Nonetheless, a nearly continuously positive skill score shows that the seasonal pattern of the observations can be simulated by the NN.

The feature importance analysis provides insight on which variables the MB-FCN-LT/ST generally relies upon. An examination of the importance of the individual branches, as shown in Figure 8, shows that, for the first forecast day, both LT and ST have a significant influence on the forecast accuracy. For longer forecast horizons, this influence decreases visibly, especially for ST. It is worth noting here that the influence of LT decreases less for Days 2–4, remaining at an almost constant level. Consequently, the long-term components of the decomposed time series have an important influence on all forecasts.

Looking at the importance of each variable with its components shows first of all that the NN is strongly dependent on the input ozone concentration. This dependence decreases continuously with lead time. Important meteorological drivers are temperature, relative humidity, and planetary boundary layer height. All these variables diminish in importance with increasing forecast horizon, analogously to the importance of the ozone concentration. On the chemical side, NO_2 also has an influence. Here, it must be emphasized that, in contrast to the other variables, the influence does not decrease with lead time, but remains constant over all forecast days. From the feature importance, we can see that the trained model does not make extensive use of information from wind, NO, or cloud cover.

Isolating the effects of the individual inputs in the LT branch shows that the NN is hardly dependent on the long-term components of the input variables apart from ozone (see Figure 9). The importance of ozone is higher on Day 1 than on the following days, but then remains at a constant level. For the short-term components, the concentration of ozone is also decisive. However, its influence decreases rapidly from the 1-day to the 2-day forecast. The individual importance of the ST components of the other input variables behaves in the same way as the overall importance of these variables.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

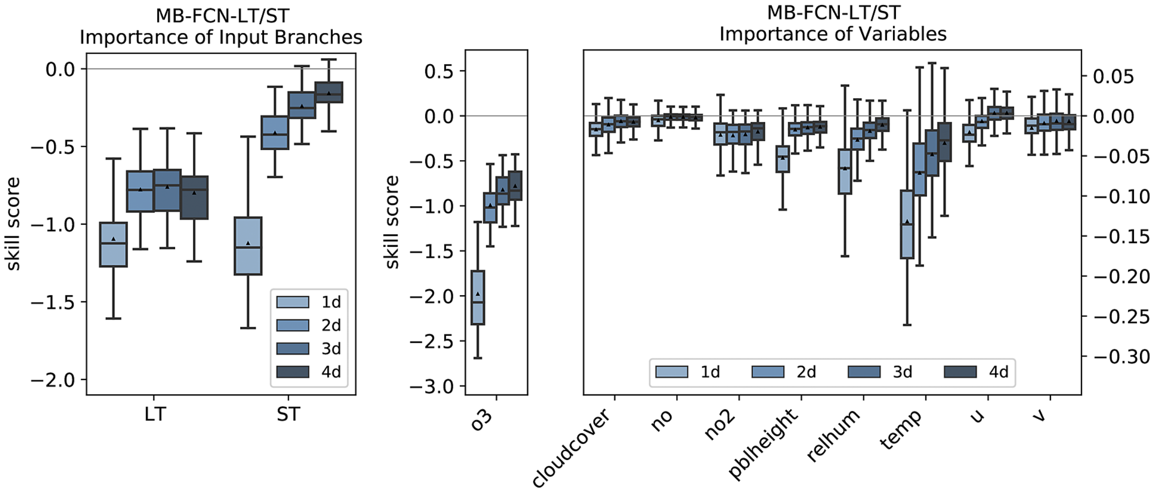


Figure 8. *Importance of single branches (left) and single variables (right) for the MB-FCN-LT/ST using bootstrapping. In blue colors, the skill score for lead times from 1 day (light blue) to 4 days (dark blue) is shown. A negative skill score indicates a strong influence on the forecast performance. The skill score is calculated with the original undisturbed prediction of the same NN as reference. Note that due to the significantly stronger dependence, ozone is visualized on a separate scale.*

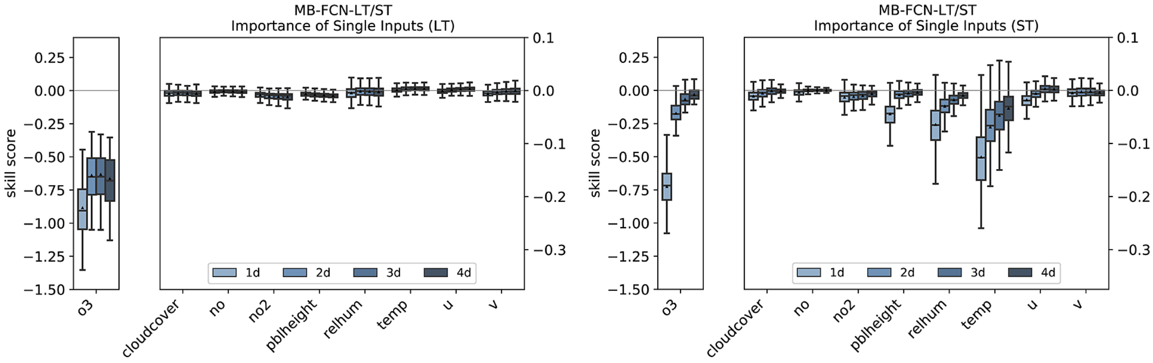


Figure 9. *Importance of single inputs for the LT branch (left) and the ST branch (right) for the MB-FCN-LT/ST using a bootstrap approach. In blue colors, the skill score for lead times from 1 day (light blue) to 4 days (dark blue) is shown. A negative skill score indicates a dependence. The skill score is calculated with the original undisturbed prediction of the same NN as reference.*

As previously mentioned, the points discussed before can be more or less transferred to the other NN architectures. The feature importance analysis of the branches and the individual variables for MB-CNN and MB-RNN is shown in Figures E1–E3 in Appendix E. In particular, the LT for all forecast days and the ST for the first day contain important information, with the ST branch being less relevant for the MB-RNN. Moreover, MB-CNN and MB-RNN also show a narrowing of the distribution of issued forecasts with increasing lead time, as was also observed for MB-FCN.

5. Discussion

The experimental results described in the previous section indicate that the NNs learn oscillation patterns on different time scales, and in particular climatological properties, better when the input time series are explicitly decomposed into different temporal components. The MB-NNs outperform all reference

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

models, such as simple statistical regression methods as well as the naïve persistence forecasts and climatological references. The MB-NNs are also preferable to their corresponding counterparts without temporal decomposition, considered individually but also as a collective.

The uncertainty estimate of the MSE of the forecast shows that FCNs that either use a decomposition into a long-term and a short-term component or access unfiltered raw data as a supplementary source of information have the highest forecast accuracy. Separating the input signals into more than two components without adding the unfiltered raw data cannot improve the performance of the FCNs, with respect to the architectures chosen in this study. This recognition coincides with the findings of Kang et al. (2013), who show that a clear separation of the short-term components is generally not possible due to the superposition of multiple oscillation patterns.

With regard to the network architecture, several key points can be identified in this study. Without special processing of the input data, the best results were achieved with a CNN architecture. This could be explained by the fact that the convolutional layers of the CNN already filter the time series. However, it must also be mentioned that with a filter size of only 5 hr, there is no chance to extract an annual cycle, so that the explicit decomposition into LT and ST components also offers added value for the CNN. However, due to the higher baseline level, the MB-CNN cannot benefit as much from the data processing compared to the MB-FCN and MB-RNN and is moreover behind the other two MB-NNs in terms of prediction quality in absolute terms. The RNN also achieves better results on the unfiltered data than the FCN, for example, because it can benefit from a more specific understanding of time. The FCN is therefore inferior to the CNN and RNN due to its comparatively simple architecture and the lack of possibility to relate neighboring data points explicitly. However, it benefits most from the temporal decomposition of the inputs, so that these disadvantages disappear, and overall, the smallest errors can be achieved with MB-FCN and MB-RNN. These finding therefore highlight the importance of jointly optimizing data preprocessing and NN model architecture, which is taught in many ML courses, but not always followed in practical applications.

The difficulties of NNs to recognize annual patterns in daily resolved data noted by Cho et al. (2014) did not apply to the MB-NNs. However, the networks still encounter difficulties in anticipating very low and very high ozone concentrations. As the lead time increases, the NN's forecast strategy becomes more cautious about extremes, leading to a narrowing of the distribution of issued forecasts. Despite this circumstance, the NNs always retain within an optimal range from a climatological point of view, so that the forecast has higher accuracy than a climatological forecast. The analysis of the feature importance can provide an explanation for this. For the first day of the forecast, both long-term and short-term components have an equally strong influence on the MB-FCN forecast, but for a longer forecast horizon, the long-term components are given more weight. Accordingly, the LT branch in particular enables the NN to generate a climatologically meaningful forecast. In addition, the NN remains strongly dependent on the ozone concentrations from the inputs. Learning a form of autocorrelation is advantageous for climatological accuracy, but at the same time leads to a poorer representation of scarcer events such as sudden and strong increases in the daily maximum concentration from one day to the next.

In addition, it must be mentioned that strong deviations from climatological norm states also have an impact on the filter decomposition of the inputs, since climatology can only be an estimate of a long-term mean state, which can deviate strongly from the actual weather in individual cases. For example, the long-term signal of temperature in the case of a very warm summer would be weakened by the added climatology, since such a deviation represents an exceptional case from a climatological point of view. In this case, the second filter component, which should actually be free of an annual variation, also contains a proportion of an annual oscillation. However, as discussed in Section 2, this combination allows to apply noncausal filters in a forecasting situation, where generally only causal filters are applicable, which lead to phase shifts in the data and show larger errors.

A look at the importance of the individual inputs for the MB-FCN yields two views. First, it becomes apparent that the dependency of the LT and ST components are each strongly based on the corresponding component of the ozone concentration and that the MB-FCN accordingly learns the connection between observed hourly ozone values and the target ozone statistic. Second, all other variables seem to have an

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

influence only on the short-term scale. Since the ST component by definition represents the deviation from the climatological normal state, it can be seen that the MB-FCN relies on the deviation from normal states as a forecasting strategy.

Finally, we would like to discuss the filters used for the decomposition. Since there are many different types of filters with various advantages and disadvantages, we have limited this work to the use of a Kaiser window and have not carried out any further experiments with different types of filters, such as the KZ filter, which could possibly lead to an improved separation of individual components as stated in Rao and Zurbenko (1994) in the presence of a weather forecast. Furthermore, we have not undertaken any in-depth investigations into which separation frequencies lead to an optimal decomposition of the time series.

6. Conclusion

In this work, we explored the potential of training different NNs, namely FCN, CNN, and RNN, for dma8eu ozone forecasting using inputs decomposed into different frequency components from long term to short term with noncausal filters in order to improve the forecast accuracy of the NNs. The temporal decomposition of the inputs not only improves the different NN architectures and the linear OLS model, but also offers an overall added value for the prediction of ozone compared to all reference models using raw hourly inputs and naïve approaches based on persistence and climatology. As exemplary shown with the MB-FCN, the MB-NNs work better with a decomposition into two components compared to four and they rely on both long-term and short-term components for their prediction, with a strong dependence on past ozone observations and a decreasing importance of the short-term components with lead time. In order to realize a valid decomposition in a forecast setup without time delay of the signal introduced by the filter itself, a combination of observations and a priori information in the form of climatology was chosen.

Acknowledgements. The authors gratefully acknowledge the Jülich Supercomputing Centre for providing the infrastructure and resources to run all experiments. We are also thankful to all air quality data providers that made their data available in the TOAR DB.

Author Contributions. Conceptualization: All authors; Data curation: L.H.L. and F.K.; Formal analysis: L.H.L.; Funding acquisition: M.G.S.; Investigation: L.H.L.; Methodology: L.H.L.; Project administration: M.G.S.; Resources: M.G.S.; Software: L.H.L. and F.K.; Supervision: M.G.S.; Validation: All authors; Visualization: L.H.L. and F.K.; Writing—original draft: L.H.L.; Writing—review and editing: All authors.

Data Availability Statement. Replication data and code can be found on b2share: https://doi.org/10.34730/decca0f4bbfc42c693812f7a648b2d6f.

Funding Statement. This research was supported by grants from the European Research Council, H2020 Research Infrastructures (Grant No. 787576 [IntelliAQ]).

Ethical Standards. The research meets all ethical guidelines, including adherence to the legal requirements of the study country.

Competing Interests. The authors declare none.

References

Bagnall A, Lines J, Bostrom A, Large J and Keogh E (2017) The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery 31*(3), 606–660.

Bauerle A, van Onzenoodt C and Ropinski T (2021) Net2Vis—a visual grammar for automatically generating publication-tailored CNN architecture visualizations. *IEEE Transactions on Visualization and Computer Graphics 27*(6), 2980–2991.

Bessagnet B, Pirovano G, Mircea M, Cuvelier C, Aulinger A, Calori G, Ciarelli G, Manders A, Stern R, Tsyro S, García Vivanco M, Thunis P, Pay M-T, Colette A, Couvidat F, Meleux F, Rouíl L, Ung A, Aksoyoglu S, Baldasano JM, Bieser J, Briganti G, Cappelletti A, D'Isidoro M, Finardi S, Kranenburg R, Silibello C, Carnevale C, Aas W, Dupont J-C, Fagerli H, Gonzalez L, Menut L, Prévôt ASH, Roberts P and White L (2016) Presentation of the EURODELTA III intercomparison exercise—evaluation of the chemistry transport models' performance on criteria pollutants and joint analysis with meteorology. *Atmospheric Chemistry and Physics 16*(19), 12667–12701.

Bollmeyer C, Keller JD, Ohlwein C, Wahl S, Crewell S, Friederichs P, Hense A, Keune J, Kneifel S, Pscheidt I, Redl S and Steinke S (2015) Towards a high-resolution regional reanalysis for the European cordex domain. *Quarterly Journal of the Royal Meteorological Society 141*(686), 1–15.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Chandra R, **Goyal S and Gupta R** (2021) Evaluation of deep learning models for multi-step ahead time series prediction. *IEEE Access 9*, 83105–83123.

Cho K, **Van Merriënboer B**, **Bahdanau D and Bengio Y** (2014) On the properties of neural machine translation: Encoder–decoder approaches. *Preprint*, arXiv:1409.1259.

Chung J, **Gulcehre C**, **Cho K and Bengio Y** (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. *Preprint*, arXiv:1412.3555.

Clevert D-A, **Unterthiner T and Hochreiter S** (2016) Fast and accurate deep network learning by exponential linear units (ELUs). *Preprint*, arXiv:1511.07289.

Cohen, A. J., Brauer, M., Burnett, R., Anderson, H. R., Frostad, J., Estep, K., Balakrishnan, K., Brunekreef, B., Dandona, L., Dandona, R., Feigin, V., Freedman, G., Hubbell, B., Jobling, A., Kan, H., Knibbs, L., Liu, Y., Martin, R., Morawska, L., … Forouzanfar, M. H. (2017). Estimates and 25-year trends of the global burden of disease attributable to ambient air pollution: an analysis of data from the Global Burden of Diseases Study 2015. *The Lancet*, *389*(10082), 1907-1918. https://doi.org/10.1016/s0140-6736(17)30505-6

Collins WJ, **Stevenson DS**, **Johnson CE and Derwent RG** (1997) Tropospheric ozone in a global-scale three-dimensional lagrangian model and its response to nox emission controls. *Journal of Atmospheric Chemistry 26*(3), 223–274.

Comrie AC (1997) Comparing neural networks and regression models for ozone forecasting. *Journal of the Air & Waste Management Association 47*(6), 653–663.

Cui Z, **Chen W and Chen Y** (2016) Multi-scale convolutional neural networks for time series classification. Preprint, arXiv: 1603.06995.

De Gooijer JG and Hyndman RJ (2006) 25 years of time series forecasting. *International Journal of Forecasting 22*(3), 443–473.

Di Q, **Dai L**, **Wang Y**, **Zanobetti A**, **Choirat C**, **Schwartz JD and Dominici F** (2017) Association of short-term exposure to air pollution with mortality in older adults. *JAMA 318*(24), 2446–2456.

Donner LJ, **Wyman BL**, **Hemler RS**, **Horowitz LW**, **Ming Y**, **Zhao M**, **Golaz J-C**, **Ginoux P**, **Lin S-J**, **Schwarzkopf MD**, **Austin J**, **Alaka G**, **Cooke WF**, **Delworth TL**, **Freidenreich SM**, **Gordon CT**, **Griffies SM**, **Held IM**, **Hurlin WJ**, **Klein SA**, **Knutson TR**, **Langenhorst AR**, **Lee H-C**, **Lin Y**, **Magi BI**, **Malyshev SL**, **Milly PCD**, **Naik V**, **Nath MJ**, **Pincus R**, **Ploshay JJ**, **Ramaswamy V**, **Seman CJ**, **Shevliakova E**, **Sirutis JJ**, **Stern WF**, **Stouffer RJ**, **Wilson RJ**, **Winton M**, **Wittenberg AT and Zeng F** (2011) The dynamical core, physical parameterizations, and basic simulation characteristics of the atmospheric component AM3 of the GFDL global coupled model CM3. *Journal of Climate 24*(13), 3484–3519.

Elkamel A, **Abdul-Wahab S**, **Bouhamra W and Alper E** (2001) Measurement and prediction of ozone levels around a heavily industrialized area: A neural network approach. *Advances in Environmental Research 5*(1), 47–59.

European Parliament and Council of the European Union (2008) Directive 2008/50/EC of the European Parliament and of the Council of 21 May 2008 on ambient air quality and cleaner air for Europe. *Official Journal of the European Union 29*, 169–212.

Fawaz HI, **Lucas B**, **Forestier G**, **Pelletier C**, **Schmidt DF**, **Weber J**, **Webb GI**, **Idoumghar L**, **Muller P-A and Petitjean F** (2020) InceptionTime: Finding AlexNet for time series classification. *Data Mining and Knowledge Discovery 34*(6), 1936–1962.

Fleming ZL, **Doherty RM**, **von Schneidemesser E**, **Malley CS**, **Cooper OR**, **Pinto JP**, **Colette A**, **Xu X**, **Simpson D**, **Schultz MG**, **Lefohn AS**, **Hamad S**, **Moolla R**, **Solberg S and Feng Z** (2018) Tropospheric Ozone Assessment Report: Present-day ozone distribution and trends relevant to human health. *Elementa: Science of the Anthropocene 6*, 12.

Fuentes M and Raftery AE (2005) Model evaluation and spatial interpolation by Bayesian combination of observations with outputs from numerical models. *Biometrics 61*(1), 36–45.

Gardner M and Dorling S (1999) Neural network modelling and prediction of hourly NO_x and NO_2 concentrations in urban air in London. *Atmospheric Environment 33*(5), 709–719.

Granier C, **Bessagnet B**, **Bond T**, **D'Angiola A**, **van Der Gon HD**, **Frost GJ**, **Heil A**, **Kaiser JW**, **Kinne S**, **Klimont Z**, *et al.* (2011) Evolution of anthropogenic and biomass burning emissions of air pollutants at global and regional scales during the 1980–2010 period. *Climatic Change 109*(1), 163–190.

Grell GA, **Peckham SE**, **Schmitz R**, **McKeen SA**, **Frost G**, **Skamarock WC and Eder B** (2005) Fully coupled "online" chemistry within the WRF model. *Atmospheric Environment 39*(37), 6957–6975.

He K, **Zhang X**, **Ren S and Sun J** (2015) Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, pp. 1026–1034.

Hilboll A, **Richter A and Burrows J** (2013) Long-term changes of tropospheric NO_2 over megacities derived from multiple satellite instruments. *Atmospheric Chemistry and Physics 13*(8), 4145–4169.

Hochreiter S and Schmidhuber J (1997) Long short-term memory. *Neural Computation 9*(8), 1735–1780.

Hornik K, **Stinchcombe M and White H** (1989) Multilayer feedforward networks are universal approximators. *Neural Networks 2*(5), 359–366.

Horowitz LW, **Stacy W**, **Mauzerall DL**, **Emmons LK**, **Rasch PJ**, **Granier C**, **Tie X**, **Lamarque J**, **Schultz MG**, **Tyndall GS**, **Orlando JJ and Brasseur GP** (2003) A global simulation of tropospheric ozone and related tracers: Description and evaluation of MOZART, version 2. *Journal of Geophysical Research: Atmospheres 108*(D24), 2–6.

Jiang G, **He H**, **Yan J and Xie P** (2019) Multiscale convolutional neural networks for fault diagnosis of wind turbine gearbox. *IEEE Transactions on Industrial Electronics 66*(4), 3196–3207.

Kaiser JF (1966) Chapter 7: Digital filters. In Kuo FF and Kaiser JF (eds), *System Analysis by Digital Computer*. New York: Wiley, pp. 218–285.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Kang D, **Hogrefe C**, **Foley KL**, **Napelenok SL**, **Mathur R and Rao ST** (2013) Application of the Kolmogorov–Zurbenko filter and the decoupled direct 3D method for the dynamic evaluation of a regional air quality model. *Atmospheric Environment 80*, 58–69.

Keren G and Schuller B (2016) Convolutional RNN: An enhanced model for extracting features from sequential data. In *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, pp. 3412–3419.

Klambauer G, **Unterthiner T**, **Mayr A and Hochreiter S** (2017) Self-normalizing neural networks. In *Advances in Neural Information Processing Systems 30*. Curran Associates Inc.

Kleinert F, **Leufen LH and Schultz MG** (2021) IntelliO3-ts v1.0: A neural network approach to predict near-surface ozone concentrations in Germany. *Geoscientific Model Development 14*(1), 1–25.

Kolehmainen M, **Martikainen H and Ruuskanen J** (2001) Neural networks and periodic components used in air quality forecasting. *Atmospheric Environment 35*(5), 815–825.

Kumar V and Sinha V (2021) Season-wise analyses of VOCs, hydroxyl radicals and ozone formation chemistry over north-west India reveal isoprene and acetaldehyde as the most potent ozone precursors throughout the year. *Chemosphere 283*, 131184.

LeCun Y, **Haffner P**, **Bottou L and Bengio Y** (1999) Object recognition with gradient-based learning. In *Shape, Contour and Grouping in Computer Vision*. Springer, Berlin, Heidelberg, pp. 319–345.

Lefohn AS, **Malley CS**, **Simon H**, **Wells B**, **Xu X**, **Zhang L and Wang T** (2017) Responses of human health and vegetation exposure metrics to changes in ozone concentration distributions in the European Union, United States, and China. *Atmospheric Environment 152*, 123–145.

Leufen LH, **Kleinert F and Schultz MG** (2021) MLAir (v1.0)—a tool to enable fast and flexible machine learning on air data time series. *Geoscientific Model Development 14*(3), 1553–1574.

Leufen LH, **Kleinert F**, **Weichselbaum F**, **Gramlich V and Schultz MG** (2022) MLAir—a tool to enable fast and flexible machine learning on air data time series, version 2.0.0, source code. Available at https://gitlab.jsc.fz-juelich.de/esde/machine-learning/mlair/-/tags/v2.0.0 (accessed 21 June 2022).

Liang M and Hu X (2015) Recurrent convolutional neural network for object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 3367–3375.

Lou Thompson M, **Reynolds J**, **Cox LH**, **Guttorp P and Sampson PD** (2001) A review of statistical methods for the meteorological adjustment of tropospheric ozone. *Atmospheric Environment 35*(3), 617–630.

Maas AL, **Hannun AY**, **Ng AY**, *et al.* (2013) Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the International Conference on Machine Learning (ICML)*, Vol. *30*. Atlanta, Georgia, USA, p. 3.

Maas R and Grennfelt P (eds) (2016) *Towards Cleaner Air. Scientific Assessment Report 2016.* EMEP Steering Body and Working Group on Effects of the Convention on Long-Range Transboundary Air Pollution, Oslo.

Manders AMM, **van Meijgaard E**, **Mues AC**, **Kranenburg R**, **van Ulft LH and Schaap M** (2012) The impact of differences in large-scale circulation output from climate models on the regional modeling of ozone and PM. *Atmospheric Chemistry and Physics 12*(20), 9441–9458.

Monks PS, **Archibald A**, **Colette A**, **Cooper O**, **Coyle M**, **Derwent R**, **Fowler D**, **Granier C**, **Law KS**, **Mills G**, *et al.* (2015) Tropospheric ozone and its precursors from the urban to the global scale from air quality to short-lived climate forcer. *Atmospheric Chemistry and Physics 15*(15), 8889–8973.

Murphy AH (1988) Skill scores based on the mean square error and their relationships to the correlation coefficient. *Monthly Weather Review 116*(12), 2417–2424.

Murphy AH (1992) Climatology, persistence, and their linear combination as standards of reference in skill scores. *Weather and Forecasting 7*(4), 692–698.

Murphy AH and Winkler RL (1987) A general framework for forecast verification. *Monthly Weather Review 115*(7), 1330–1338.

Oppenheim AV and Schafer RW (1975) *Digital Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall.

Ortiz A and Guerreiro C (2020) *Air Quality in Europe—2020 Report.* European Environment Agency, Publications Office, Copenhagen, Denmark.

Rao S, **Zurbenko I**, **Neagu R**, **Porter P**, **Ku J and Henry R** (1997) Space and time scales in ambient ozone data. *Bulletin of the American Meteorological Society 78*(10), 2153–2166.

Rao ST and Zurbenko IG (1994) Detecting and tracking changes in ozone air quality. *Air & Waste 44*(9), 1089–1092.

Reichstein M, **Camps-Valls G**, **Stevens B**, **Jung M**, **Denzler J**, **Carvalhais N**, *et al.* (2019) Deep learning and process understanding for data-driven earth system science. *Nature 566*(7743), 195–204.

REVIHAAP (2013) *Review of Evidence on Health Aspects of Air Pollution—REVIHAAP Project Technical Report.* Bonn: World Health Organization (WHO) Regional Office for Europe.

Richter A, **Burrows JP**, **Nüß H**, **Granier C and Niemeier U** (2005) Increase in tropospheric nitrogen dioxide over China observed from space. *Nature 437*(7055), 129–132.

Russell A, **Valin L and Cohen R** (2012) Trends in OMI NO_2 observations over the United States: Effects of emission control technology and the economic recession. *Atmospheric Chemistry and Physics 12*(24), 12197–12209.

Schultz MG, **Betancourt C**, **Gong B**, **Kleinert F**, **Langguth M**, **Leufen LH**, **Mozaffari A and Stadtler S** (2021) Can deep learning beat numerical weather prediction? *Philosophical Transactions of the Royal Society A 379*(2194), 20200097.

Schultz MG, **Schröder S**, **Lyapina O**, **Cooper OR**, **Galbally I**, **Petropavlovskikh I**, **Von Schneidemesser E**, **Tanimoto H**, **Elshorbany Y**, **Naja M**, *et al.* (2017). Tropospheric Ozone Assessment Report: Database and metrics data of global surface ozone observations. *Elementa: Science of the Anthropocene 5*, 58.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Seltzer KM, Shindell DT, Faluvegi G, & Murray LT (2017). Evaluating Modeled Impact Metrics for Human Health, Agriculture Growth, and Near-Term Climate. *Journal of Geophysical Research: Atmospheres*, *122*(24), 13, 506–13, 524. Portico. https://doi.org/10.1002/2017jd026780

Seltzer KM, Shindell DT, Kasibhatla P and Malley CS (2020) Magnitude, trends, and impacts of ambient long-term ozone exposure in the United States from 2000 to 2015. *Atmospheric Chemistry and Physics 20*(3), 1757–1775.

Shih S-Y, Sun F-K and Lee H (2019) Temporal pattern attention for multivariate time series forecasting. *Machine Learning 108*(8), 1421–1441.

Shindell D, Faluvegi G, Kasibhatla P and Van Dingenen R (2019) Spatial patterns of crop yield change by emitted pollutant. *Earth's Future 7*(2), 101–112.

Simon H, Reff A, Wells B, Xing J and Frank N (2015) Ozone trends across the United States over a period of decreasing NO_x and VOC emissions. *Environmental Science & Technology 49*(1), 186–195.

Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V and Rabinovich A (2015) Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1–9.

TOAR (2019). *Tropospheric Ozone Assessment Report (TOAR): Global Metrics for Climate Change, Human Health and Crop/Ecosystem Research.* International Global Atmospheric Chemistry. Available at https://igacproject.org/activities/TOAR (accessed 29 January 2021).

US EPA (2013) *Integrated Science Assessment (ISA) for Ozone and Related Photochemical Oxidants (Final Report, February 2013)*. U.S. Environmental Protection Agency, Washington, DC, EPA/600/R-10/076F.

US EPA (2020) *Integrated Science Assessment (ISA) for Ozone and Related Photochemical Oxidants (Final Report, April 2020)*. U.S. Environmental Protection Agency, Washington, DC, EPA/600/R-20/012.

Vautard R, Moran MD, Solazzo E, Gilliam RC, Matthias V, Bianconi R, Chemel C, Ferreira J, Geyer B, Hansen AB, Jericevic A, Prank M, Segers A, Silver JD, Werhahn J, Wolke R, Rao S and Galmarini S (2012) Evaluation of the meteorological forcing used for the air quality model evaluation international initiative (AQMEII) air quality simulations. *Atmospheric Environment 53*, 15–37. AQMEII: An International Initiative for the Evaluation of Regional-Scale Air Quality Models—Phase 1.

von Kuhlmann R, Lawrence MG, Crutzen PJ and Rasch PJ (2003) A model for studies of tropospheric ozone and nonmethane hydrocarbons: Model description and ozone results. *Journal of Geophysical Research: Atmospheres 108*(D9), 4294.

Wang Y, Jacob DJ and Logan JA (1998a) Global simulation of tropospheric O_3-NO_x-hydrocarbon chemistry: 1. Model formulation. *Journal of Geophysical Research: Atmospheres 103*(D9), 10713–10725.

Wang Y, Logan JA and Jacob DJ (1998b) Global simulation of tropospheric O_3-NO_x-hydrocarbon chemistry: 2. Model evaluation and global ozone budget. *Journal of Geophysical Research: Atmospheres 103*(D9), 10727–10755.

Wilks DS (2006) *Statistical Methods in the Atmospheric Sciences*, 2nd Edn. London: Academic Press.

Wise EK and Comrie AC (2005) Extending the Kolmogorov–Zurbenko filter: Application to ozone, particulate matter, and meteorological trends. *Journal of the Air & Waste Management Association 55*(8), 1208–1216.

Young PJ, Naik V, Fiore AM, Gaudel A, Guo J, Lin MY, Neu JL, Parrish DD, Rieder HE, Schnell JL, Tilmes S, Wild O, Zhang L, Ziemke J, Brandt J, Delcloo A, Doherty RM, Geels C, Hegglin MI, Hu L, Im U, Kumar R, Luhar A, Murray L, Plummer D, Rodriguez J, Saiz-Lopez A, Schultz MG, Woodhouse MT and Zeng G (2018) Tropospheric Ozone Assessment Report: Assessment of global-scale model performance for global and regional ozone distributions, variability, and trends. *Elementa: Science of the Anthropocene 6*, 10.

Zhang Y, Cooper OR, Gaudel A, Thompson AM, Nédélec P, Ogino S-Y and West JJ (2016) Tropospheric ozone change from 1980 to 2010 dominated by equatorward redistribution of emissions. *Nature Geoscience 9*(12), 875–879.

Zhang Y, West JJ, Mathur R, Xing J, Hogrefe C, Roselle SJ, Bash JO, Pleim JE, Gan C-M and Wong DC (2018) Long-term trends in the ambient $PM_{2.5}$- and O_3-related mortality burdens in the United States under emission reductions from 1990 to 2010. *Atmospheric Chemistry and Physics 18*, 1–14.

Zhao J, Huang F, Lv J, Duan Y, Qin Z, Li G and Tian G (2020) Do RNN and ISTM have long memory? In *Proceedings of the 37th International Conference in Machine Learning*. Proceedings of Machine Learning Research (PMLR) *119*, online, pp. 11365–11375.

Zheng Y, Liu Q, Chen E, Ge Y and Zhao JL (2014) Time series classification using multi-channels deep convolutional neural networks. In Li F, Li G, Hwang S-w, Yao B and Zhang Z (eds), *Web-Age Information Management*. Cham: Springer International Publishing, pp. 298–310.

Ziyin L, Hartwig T and Ueda M (2020) Neural networks fail to learn periodic functions and how to fix it. In Larochelle H, Ranzato M, Hadsell R, Balcan M and Lin H (eds), *Advances in Neural Information Processing Systems*, Vol. *33*. Curran Associates, Inc., pp. 1583–1594.

Žurbenko IG (1986) *The Spectral Analysis of Time Series*, Vol. *2*. North-Holland Series in Statistics and Probability. Elsevier, Amsterdam.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Appendix A: Details on Data

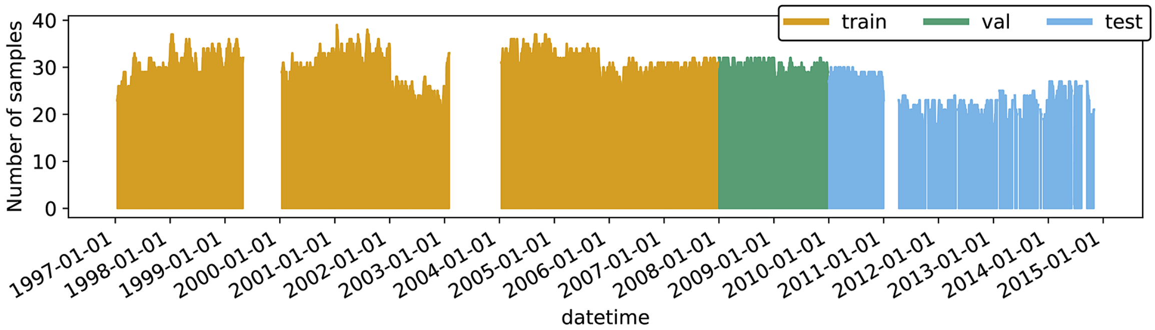


Figure A1. *Graphical representation of the number of samples available for training (orange), validation (green), and testing (blue) per time step. Apart from three periods in which the data cannot meet the requirements, more than 20 stations are available at each time step, and for training in particular, more than 30 stations for the most time. The graph does not show the available raw data, but indicates for which time steps t_0 a sample with fully processed input and target values is available.*

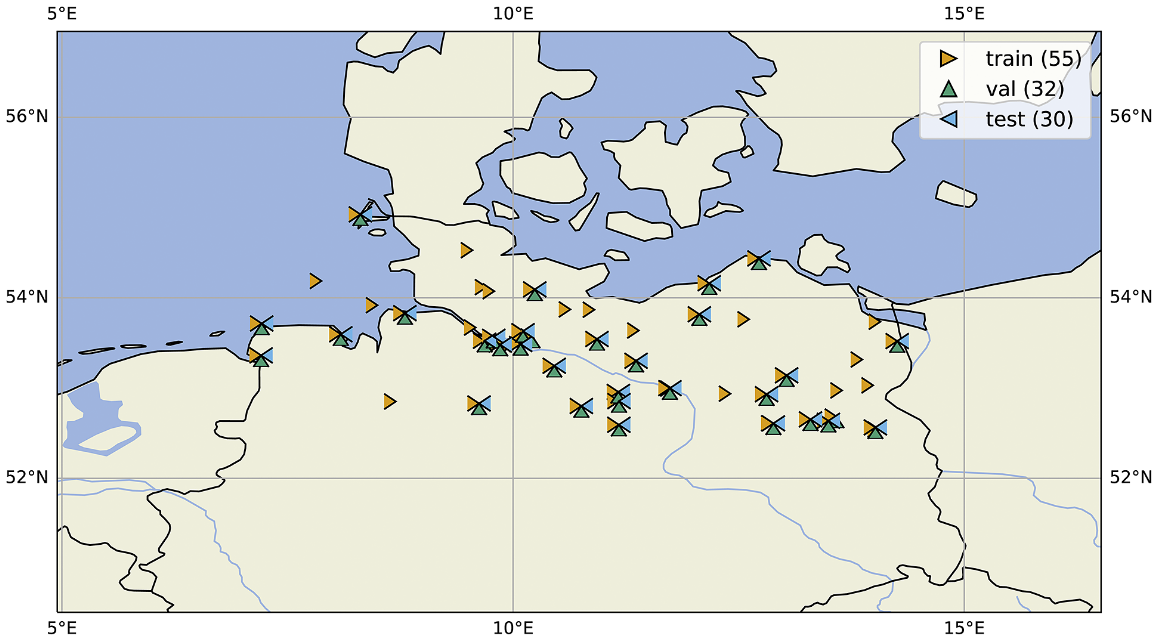


Figure A2. *Geographical location of all rural and suburban monitoring stations used in this study divided into training (orange), validation (green), and test (blue) data represented by triangles in the corresponding colors. The tip of the triangles points to the exact location of the station.*

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

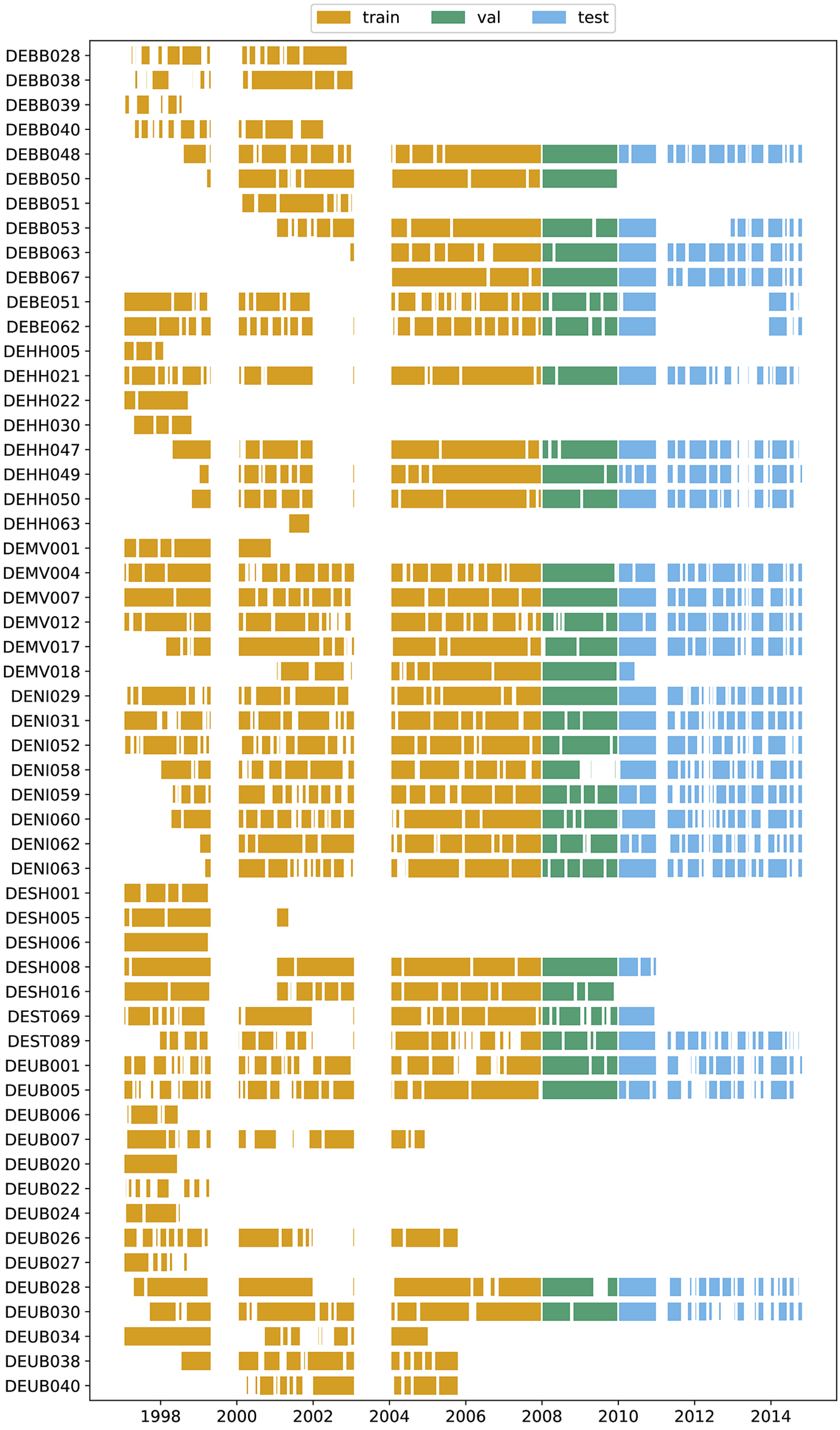


Figure A3. *Detailed overview of the availability of station data broken down for all individual stations as a timeline separated by color for training (orange), validation (green), and test (blue) data. Individual gaps are caused by missing observation data that exceed the interpolation limit of 24 hr for inputs or 2 days for targets.*

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Appendix B: Details on Hyperparameter Search

Table B1. Details on tested hyperparameters for the MB-FCNs. The square brackets indicate a continuous parameter range, and the curly brackets indicate a fixed set of parameters. Parameter spaces covering different orders of magnitude were sampled on a logarithmic scale. For details on the activation functions, we refer to rectified linear unit (ReLU) and leaky rectified linear unit (LeakyReLU, Maas et al., 2013), exponential linear unit (ELU, Clevert et al., 2016), scaled exponential linear unit (SELU, Klambauer et al., 2017), and parametric rectified linear unit (PReLU, He et al., 2015).

Parameter	Parameter range
Learning rate	$[0.1, 0.0001]$
Learning rate decay	$[0, 0.0001]$
Batch size	$\{64, 128, 256, 512\}$
Activation function	$\{\mathrm{relu}, \mathrm{leakyrelu}, \mathrm{elu}, \mathrm{selu}, \mathrm{prelu}\}$
Dropout	$[0, 0.5]$
Batch normalization	$\{\mathrm{true}, \mathrm{false}\}$
Branch layers	$\{512/256/128, 512/128/32, 512/64, 512/32, 256/128/64/32, 256/64, 128/64, 128/32, 64/32\}$
Tail layers	$\{4, 32/4, 64/4\}$

Table B2. Summary of best hyperparameters and fixed parameters for different setups with MB-FCN. The entire parameter ranges of all hyperparameters are given in Table B1. Details on the activation functions can be found in He et al. (2015) for the parametric rectified linear unit (PReLU) and in Clevert et al. (2016) for the exponential linear unit (ELU). A visualization of MB-FCN-LT/ST can be found in Figure D1 in addition.

Parameter	MB-FCN-BL/SY/DU/ID	MB-FCN-LT/ST	MB-FCN-BL/SY/DU/ID+raw	MB-FCN-LT/ST+raw
Hyperparameters				
Learning rate	0.00033	0.1	0.00027	0.0002
Learning rate decay	0.001	0.007	0.0001	0.0002
Batch size	512	512	512	256
Activation function	PReLU	ELU	ELU	ELU
Dropout	0.3	0.56	0.28	0.43
Batch normalization	True	True	True	True
Branch layers	64/32	128/64	64/32	128/64
Tail layers	64/4	4	4	4
Layers summary	4x(585/64/32)-64/4	2x(585/128/64)-4	5x(585/64/32)-4	3x(585/128/64)-4
Trainable parameters	168,196	167,812	199,524	251,716
Fixed				
Cutoff period(s)	21 days, 2.7 days, 11 hr	21 days	21 days, 2.7 days, 11 hr	21 days
Filter order(s)	42 days, 7 days, 2 days	42 days	42 days, 7 days, 2 days	42 days
filter window	Kaiser ($\beta = 5$)	Kaiser ($\beta = 5$)	Kaiser ($\beta = 5$)	Kaiser ($\beta = 5$)
Use unfiltered raw inputs	False	False	True	True
Number of epochs[a]	150	150	150	150

Abbreviation: FCN, fully connected network.

[a] With early stopping.

Table B3. Summary of best hyperparameters and fixed parameters for experiments with the CNN, MB-CNN, RNN, and MB-RNN. The entire parameter ranges of all hyperparameters are not listed. Details on the activation functions can be found in Maas et al. (2013) for the rectified linear unit (ReLU) and the leaky rectified linear unit (LeakyReLU) and in He et al. (2015) for the parametric rectified linear unit (PReLU).

Parameter	CNN	MB-CNN	RNN	MB-RNN
Hyperparameters				
Learning rate	0.057	0.1668	0.0009	0.0123
Learning rate decay	0.006	0.009	0.0006	0.015
Activation function	PReLU	PReLU	ReLU	LeakyReLU
Dropout	0.43	0.42	0.5 & 0.17 (recurrent)	0.23 & 0 (recurrent)
Batch normalization	Conv and FC	Conv and FC	only LSTM	only LSTM
Filter size	5 × 1	5 × 1	–	–
(Branch) layers[a]	C16/MP/C32/MP/C64	C16/MP/C32/MP/C64	LSTM64	LSTM32
Tail/dense layers	256/4	256/4	128/4	128/4
Trainable parameters	281,140	560,228	27,908	19,716
Fixed				
Number of epochs[b]	250	250	100	100
Batch size	512	512	512	512

Abbreviations: CNN, convolutional neural network; LSTM, long short-term memory.

[a] C<*n*>: Conv2D with *n* filters; LSTM<*n*>: LSTM layer with *n* LSTM cells; MP: MaxPooling.

[b] With early stopping.

Appendix C: Tabular Results

Table C1. Key numbers of the uncertainty estimation of the MSE for all MB-FCNs as an average over all prediction days using the bootstrap approach visualized in Figure 3. All reported numbers are in the unit of square parts per billion. Numbers in percentage point to the corresponding percentile of the error distribution.

	MB-FCN-BL/ SY/DU/ID	MB-FCN-BL/SY/ DU/ID+raw	MB-FCN-LT/ST	MB-FCN-LT/ST+raw	FCN
Mean	71.83	67.88	67.12	**66.72**	77.51
Min	56.56	**55.15**	57.38	56.17	67.01
Lower whisker	59.15	56.41	57.38	**56.17**	68.22
25%	68.53	64.92	64.25	**63.70**	75.25
50%	71.67	67.72	66.99	**66.54**	77.42
75%	74.78	70.59	69.69	**69.40**	79.94
Higher whisker	84.16	79.09	**77.86**	77.93	86.97
Max	87.52	82.17	80.89	**80.59**	91.75

Abbreviations: FCN, fully connected network; MSE, mean square error.

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

Table C2. Key numbers of the uncertainty estimation of the MSE as an average over all prediction days using the bootstrap approach visualized in Figure 5. All reported numbers are in the unit of square parts per billion. Numbers in percentage point to the corresponding percentile of the error distribution. Note that the uncertainty estimation reported here is independent of the results shown in Table C1, and therefore numbers may vary for statistical reasons.

	CNN	FCN	IntelliO3	MB-CNN-LT/ST	MB-FCN-LT/ST	MB-RNN-LT/ST	OLS-LT/ST	OLS	Persistence	RNN
Mean	71.94	78.02	74.59	67.28	66.41	**66.08**	67.84	72.41	107.89	72.26
Min	41.93	50.74	40.75	39.85	38.52	**38.12**	40.03	40.87	52.66	42.49
25%	60.52	69.90	62.47	57.43	57.20	**55.55**	58.85	59.99	83.80	61.88
50%	75.66	80.53	77.36	70.27	**69.33**	69.56	71.09	76.93	115.17	75.38
75%	82.32	86.49	85.63	76.75	**75.44**	75.77	76.95	83.40	130.46	82.02
Max	105.92	107.01	121.41	101.99	98.55	99.63	**98.38**	104.31	168.47	105.20

Abbreviations: CNN, convolutional neural network; FCN, fully connected network; MSE, mean square error; OLS, least squares regression.

Appendix D: Model Architecture

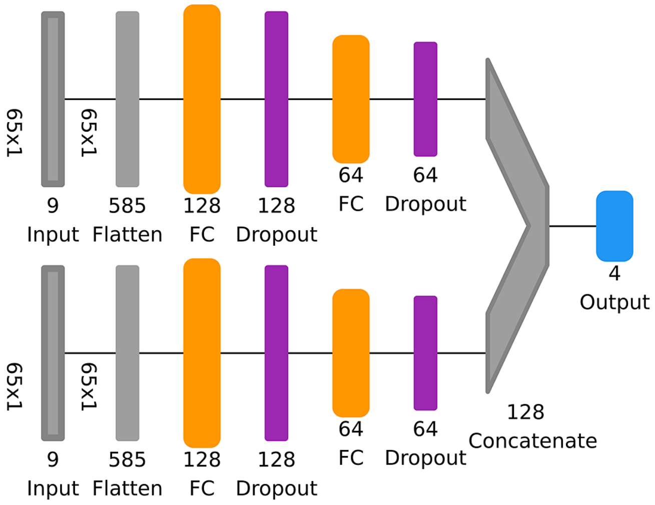


Figure D1. *Visualization of MB-FCN-LT/ST using the tool Net2Vis (Bauerle et al., 2021). Shown from left to right are the input data, followed by the flattened layer and two fully connected layers (FC) with 128 and 64 neurons. In total, the neural network has two such branches, whose weights can be trained independently of each other. All branches are concatenated and bounded by the output layer with four neurons. The orange FC block consists of a fully connected layer, a batch normalization layer, and an exponential linear unit activation. The output layer contains only a fully connected layer followed by a linear activation. The dropout layers are highlighted in purple, and all other remaining layers with nontrainable parameters are shown in gray.*

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

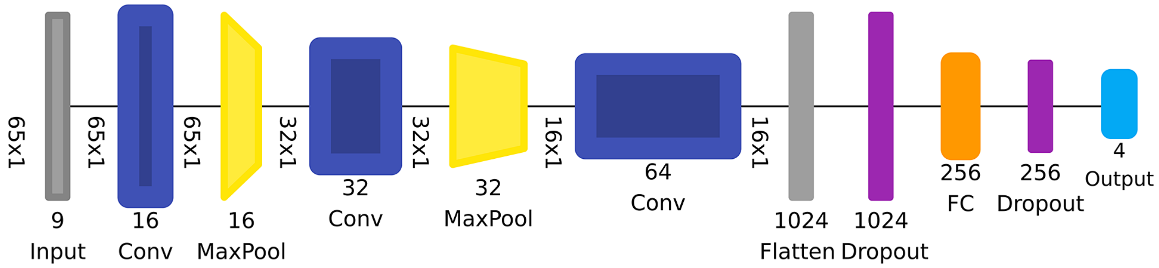


Figure D2. *Visualization of a convolutional neural network as in Figure D1. In addition, this neural network consists of convolutional blocks highlighted in blue and MaxPooling layers shown in yellow. Each convolutional block consists of a convolutional layer with a kernel size of 5 × 1 and the same padding, followed by a batch normalization layer and a parametric rectified linear unit (PreLU) activation. The MaxPooling layers use a pooling size of 2 × 1 and strides with 2 × 1. The FC blocks in this model consist of the fully connected layer, batch normalization, and a PReLU activation.*

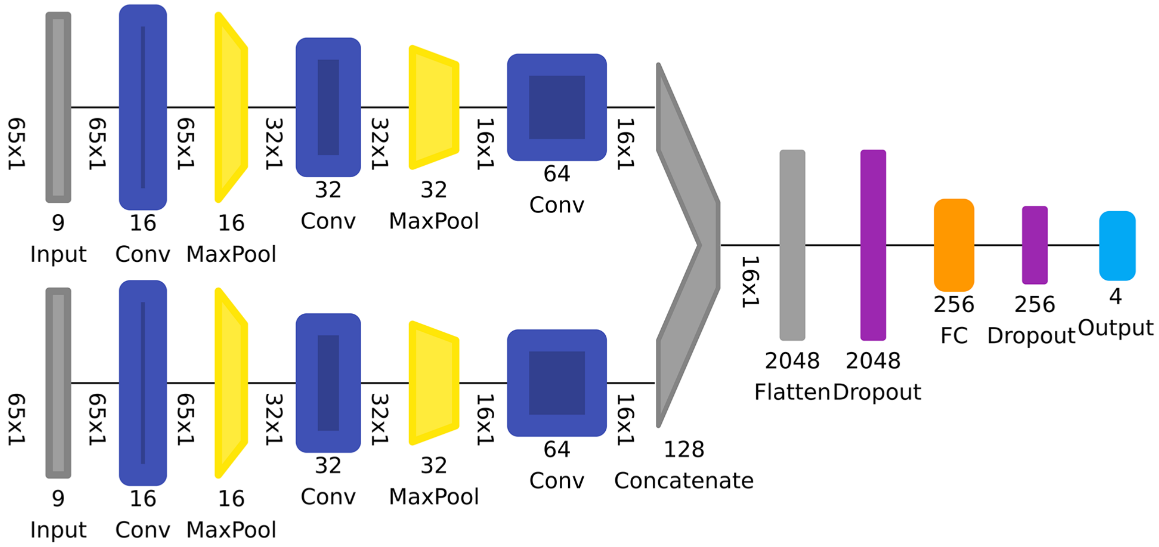


Figure D3. *Visualization of a multibranch convolutional neural network as in Figure D2.*

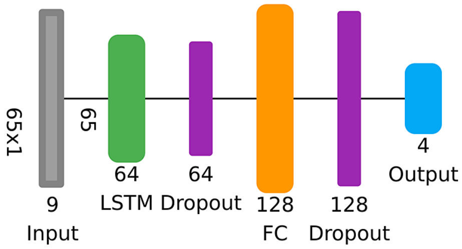


Figure D4. *Visualization of RNN as in Figure D1. In addition, the neural network shown here consists of long short-term memory layer (LSTM) blocks indicated in green. Each LSTM block includes an LSTM layer with a given number of LSTM cells within followed by a batch normalization layer and a rectified linear unit (ReLU) activation function. Note that the dropout shown here is not the recurrent dropout, but the regular dropout that is applied on the activation of a layer. The FC block also uses a ReLU activation function, but no batch normalization.*

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

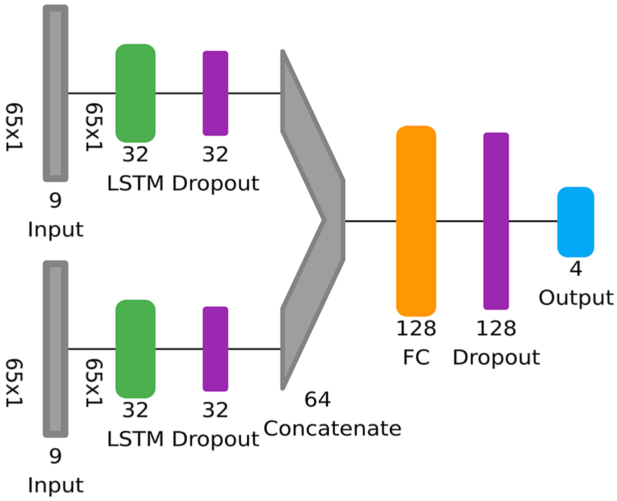


Figure D5. *Visualization of MB-RNN as in Figure D4. Deviating here, the activation is LeakyReLU both for the long short-term memory layer and the FC layer.*

Appendix E: Feature Importance of MB-CNN and MB-RNN

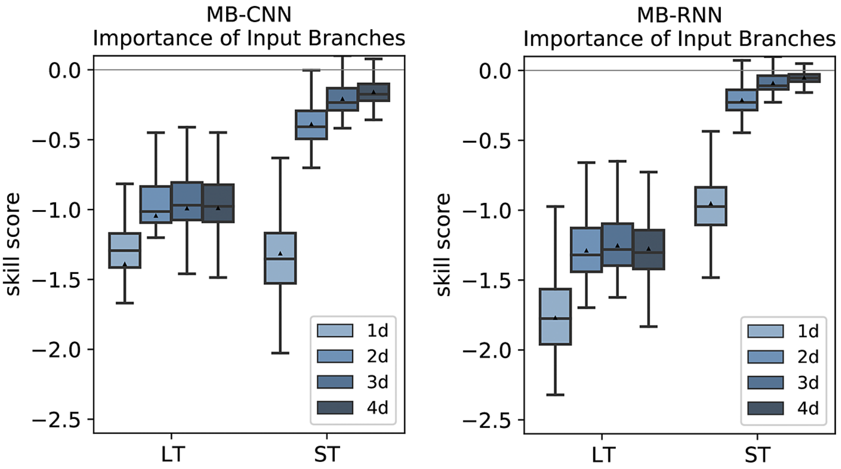


Figure E1. *Importance of single branches for multibranch convolutional neural network (left) and multibranch recurrent neural network (right) as in Figure 8.*

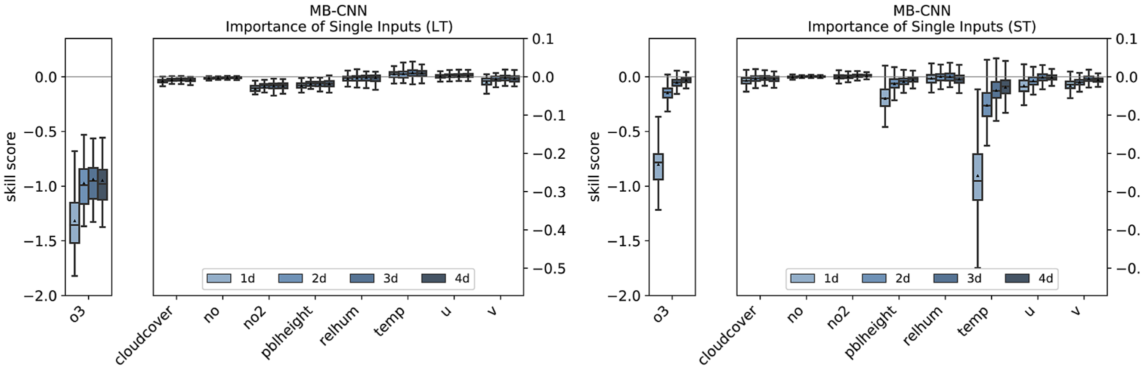


Figure E2. *Importance of single inputs for the LT branch (left) and the ST branch (right) for the multibranch convolutional neural network.*

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

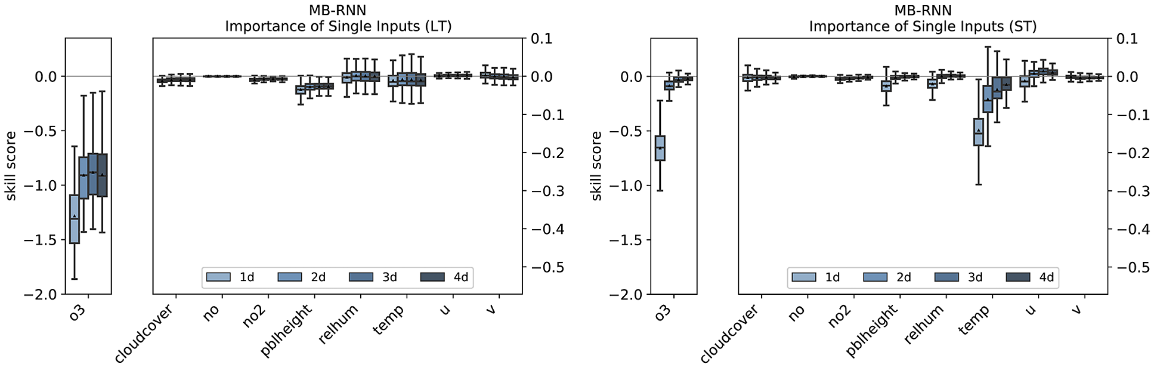


Figure E3. *Importance of single inputs for the LT branch (left) and the ST branch (right) for the multibranch recurrent neural network.*

Cite this article: Leufen LH, Kleinert F. and Schultz MG (2022). Exploring decomposition of temporal patterns to facilitate learning of neural networks for ground-level daily maximum 8-hour average ozone prediction. *Environmental Data Science*, 1: e10. doi:10.1017/eds.2022.9

https://doi.org/10.1017/eds.2022.9 Published online by Cambridge University Press

D.3. Leufen et al. (under review 2023): *O3ResNet: A deep learning based forecast system to predict local ground-level daily maximum 8-hour average ozone in rural and suburban environment*

O3ResNet: A deep learning based forecast system to predict local ground-level daily maximum 8-hour average ozone in rural and suburban environment

Lukas Hubert Leufen
Jülich Supercomputing Centre
Research Centre Jülich
Jülich, Germany
l.leufen@fz-juelich.de

Felix Kleinert
Jülich Supercomputing Centre
Research Centre Jülich
Jülich, Germany

Martin G. Schultz
Jülich Supercomputing Centre
Research Centre Jülich
Jülich, Germany

March 3, 2023

Abstract

With the impact of tropospheric ozone pollution on humankind, there is a compelling need for robust air quality forecasts. Here, we introduce a novel deep learning (DL) forecasting system called O3ResNet that produces a four-day forecast for ground-level ozone. O3ResNet is based on a convolutional neural network with residual blocks. The model has been trained on 22 years of ozone and nitrogen oxides in-situ measurements and ERA5 reanalysis data from 2000 to 2021 at 328 stations in Central Europe located in rural and suburban environment. Our model outperforms the state-of-the-art Copernicus Atmosphere Monitoring Service regional forecast model ensemble for ground-level ozone with respect to the mean square error and mean absolute error of the daily maximum 8-hour running average ozone, thus marking a major milestone for DL-based ozone prediction. O3ResNet has a very small bias without requiring additional post-processing, and it generalizes well so that new stations can be added with no need to re-train the neural network. As the model works on hourly data, it can be easily adapted to output other air quality metrics. We conclude that O3ResNet is sufficiently advanced and robust to become a test application for operational air quality forecasting with DL.

Keywords Forecasting · Neural networks · Air quality · Ozone · Deep learning · Machine learning

1 Introduction

Data-driven methods like machine learning (ML) and in particular deep learning (DL) have the potential to replace or augment classical environmental modelling approaches, because they can learn complex, intrinsic relationships among observed variables, and because they exhibit small bias by design [1]. Especially at small local scales, atmospheric phenomena are often not well described by existing theories and classical model predictions are therefore imprecise. As a complex interplay of meteorology, chemistry, emissions and landscape is involved [2], this is especially critical for ozone air pollution.

Exposure to ozone has a damaging effect on terrestrial life forms [3, 4, 5]. In particular, exposure to high ozone concentrations leads to adverse health effects in humans, especially in the pulmonary and cardiovascular systems [6]. Short-term exposure to high ozone concentrations has drastic effects [7, 8, 9], such as reduced lung function or triggering of asthma. Consequently, it is important to have reliable predictions of ozone concentrations several days in advance, in order to initiate appropriate countermeasures where necessary. Regulatory authorities around the world therefore define target and limit values for ozone. These are typically based on the daily maximum 8-hour running average [dma8, 6], so the analysis and prediction of dma8 ozone is a task of high societal relevance.

Current forecast models are based on chemistry transport models (CTMs) built on chemical and physical relationships and equations to calculate air quality numerically. However, in such models, uncertainties arise due to various causes, such as parameterizations, simplification of relationships and equations or other assumptions [10, 11, 12, 13]. These, in turn, lead to systematic deviations between the model results and the observations [14]. For example, the seasonal cycle of ozone is not well represented by the CTMs, nor do they capture the sensitivity of the models to meteorological drivers relevant for ozone formation and removal processes such as solar radiation and relative humidity well [14]. Also, CTMs are too coarse-scale to resolve local phenomena [15] and they impose a substantial computational burden for solving chemical equations [16], which is critical when deployed operationally, where wall-clock time is a hard constraint [17].

To enable DL methods to learn how to reliably predict ozone concentrations, one needs to apply domain knowledge for constructing the input data and the DL model. Temperature has an important influence on ozone, as chemical reactions are generally temperature-dependent [11]. In particular, extreme ozone concentrations are mainly linked to high temperature periods [18, 19]. Besides, persistence is also a strong predictor of high ozone levels as it can indicate the presence of prolonged events and those with a day-by-day increase in concentrations [20]. Further meteorological factors that influence local ozone levels are solar radiation and cloud cover, as well as relative humidity and wind speed [19]. Weng et al. [21], based on random forest and ridge regression, identify, for example, temperature, surface solar radiation downward, and relative humidity as the key meteorological drivers of ozone. Their study also reveals, however, that the importance of individual variables can vary between different regions. Recent studies have shown that neural networks (NNs) are skillful methods for ozone forecasting purposes and a variety of NN architectures have been explored in this context. For example, [22] use fully connected networks (FCNs), [23] convolutional neural networks (CNNs), [24] CNNs with inception blocks, [25] long short-term memory networks (LSTMs) or [26] and [27] U-Nets. However, to the best of our knowledge, there has been no study on forecasting of ozone at station locations that reports both good performance for lead times greater than two days and provides a direct comparison with a state-of-the-art CTM.

This paper presents the development of a generic DL-based ozone forecasting system called O3ResNet, that is based on a CNN architecture with residual blocks [28], to forecast ground-level dma8 ozone at individual stations. To showcase O3ResNet, we selected 328 stations in rural and suburban areas across Central Europe for study, though the system can easily be adapted to other regions, provided that enough training data is available. Results of O3ResNet are more accurate than the Copernicus Atmosphere Monitoring Service (CAMS) regional ensemble forecast [29], which is the state-of-the-art air quality forecast system in Europe. Therefore, O3ResNet provides a reliable dma8 ozone forecast for the next four days, hereafter denoted D1 to D4, which makes it a tool that is suitable for operational air quality forecasting.

This paper begins with a description of the data and methods used, followed by the results section, in which we draw a comparison to CAMS in addition to evaluating the overall performance of our model. Here, we also provide insights about the dependence of O3ResNet on its inputs and the lead time of a meteorological forecast. The paper concludes with a discussion of various aspects of O3ResNet including a consideration of the benefits and limitations of O3ResNet as well as thoughts on a roadmap towards operational deployment and the extension to forecasting other air pollutants.

2 Data and methods

Data O3ResNet has been trained with data from 328 observation stations over central Europe (47.5°-56°N and 1.3°-18°E, see Figure 1). We make use of the tropospheric ozone assessment report database [TOAR DB, 30] and select all stations located in a rural or suburban environment and classified by the European Environmental Agency as background stations [31]. This means that there is no dominant air pollution source in the immediate vicinity. To prevent temporal data leakage, data are divided block-wise along the time axis into training (2000-2015), validation (2016-2018) and test (2019-2021) data. Further details on the data split and a robustness analysis are presented in Appendix A. Due to missing or terminated observations, the number of stations varies for the validation (212) and test (202) subsets. In total, there are over 800,000 training samples, almost 200,000 for validation and 170,000 samples for testing, respectively.

Inputs For inputs, O3ResNet makes use of hourly time series of three chemical and seven meteorological variables at or near ground-level: ozone (O_3), nitric oxide (NO), nitrogen dioxide (NO_2), cloud cover, planetary boundary layer height, pressure, relative humidity, temperature and the zonal and meridional wind components. Relative humidity is calculated from temperature, dew point temperature and pressure. The selection of these parameters is based on previous research in Leufen et al. [32], so we do not apply any new feature selection here. Chemical parameters (O_3, NO, NO_2) are provided by the TOAR DB, and meteorological variables originate from ECMWF's ERA5 reanalysis data set [33], with grid data mapped to station locations using nearest neighbor interpolation. All time series are

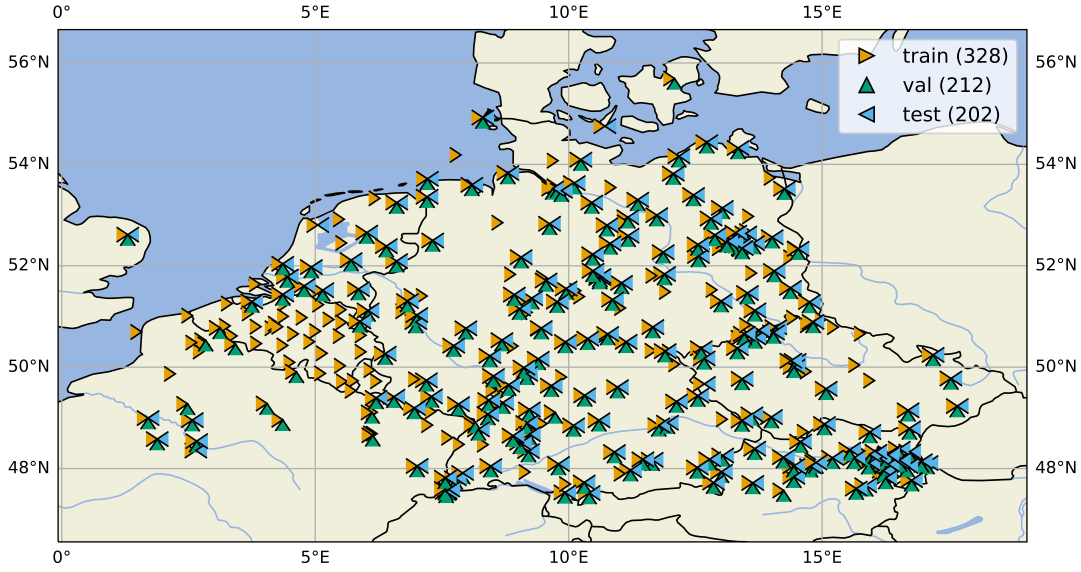


Figure 1: Geographic overview of the ozone measurement stations in Central Europe (47.5°-56°N and 1.3°-18°E). Of the total 328 stations available for the 2000-2021 period, all 328 stations were used for training O3ResNet (represented by orange triangles with apex oriented to the right). There are 212 stations available for validation (green triangle with apex oriented up) and 202 stations available for final testing (blue triangles, apex oriented left). The differences in the station numbers result from the data availability in the TOAR DB.

filtered into long-term (LT) and short-term (ST) components with a finite impulse response (FIR) filter as in [32]. For causality reasons, we use the observations for lagged time steps ($t_i \leq t_0$) and climatology for time steps in the future ($t_i > t_0$) to calculate LT and ST of the chemical variables, as proposed in [32]. For the meteorological variables we use reanalysis data as pseudo-forecast for all time steps t_i. A more detailed discussion on the time filtering can be found in Appendix A. For the chemical inputs, we choose time steps of past three days (72 hours) from LT and ST components ($[t_0 - 3d, t_0]$), the meteorological components cover in addition the forecast period on the interval of $[t_0 - 3d, t_0 + 4d]$ with a total of 168 hourly values. All inputs are transformed by Z-score normalization to have zero mean and a standard deviation of one.

Target The target variable of this study is dma8 ozone as defined by the [31] as the highest 8-hour moving average of all ozone concentrations observed between 5 pm local time of the previous day and 4 pm local time of the current day. We predict dma8 ozone for the next four days ($[t_0 + 1d, t_0 + 4d]$). The daily resolved dma8 ozone for the model validation is obtained directly from TOAR DB. The temporal distribution of the target values in all subsets is shown in Figure 13 in Appendix A. Like the inputs, the targets are transformed by Z-score normalization. Figure 2 provides an overview of the entire workflow.

Hyperparameter tuning We test different architectures like FCN, recurrent NN (RNN) based on LSTM and gated recurrent unit (GRU), CNN (with and without residual blocks) and U-Net. To find an optimal hyperparameter configuration for each NN architecture, we train NNs with various configurations over 100 epochs and evaluate the mean squared error (MSE) given by

$$\textbf{MSE} = \frac{1}{n_i \cdot n_j} \sum_{i,j}^{n_i,n_j} \left(y_{i,j} - \hat{y}_{i,j}\right)^2 \tag{1}$$

on the training and validation data with n_i being the number of samples, n_j the number of forecast steps, $y_{i,j}$ the observed value and $\hat{y}_{i,j}$ the NN's forecast. After testing all alternative model architectures, we chose a CNN architecture with residual blocks as the best performing on validation data. In Appendix B, we present details on the hyperparameter optimization and model selection strategies, and provide technical background on the operating system, software, and the duration of preprocessing, training, and inference.

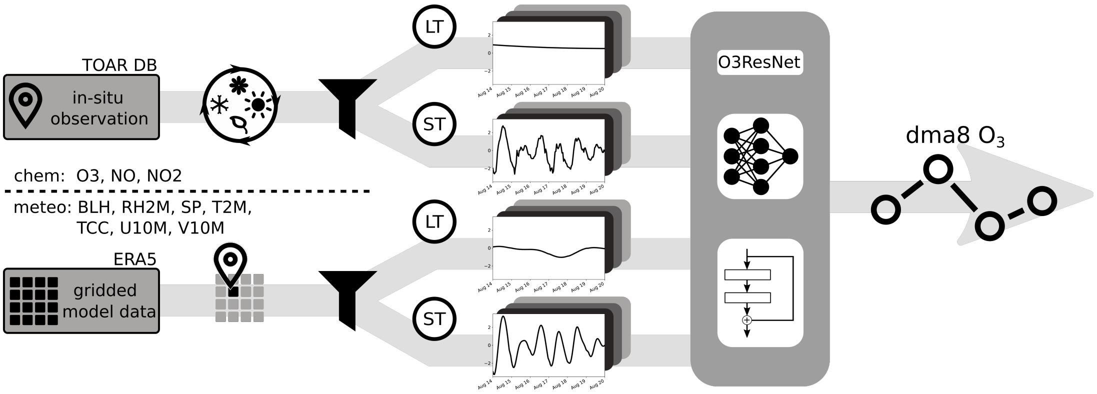


Figure 2: Visualization of the training and inference workflow of O3ResNet as described in this paper. The chemical variables are taken from the tropospheric ozone assessment report database [TOAR DB, 30] as in-situ observations and filtered into LT and ST components with the help of climatological statistics. The meteorological variables are obtained as gridded data from ERA5 and are mapped to the measurement stations by nearest neighbor and also split into LT and ST by filter. Note that variable names are listed according to the identifiers in the official documentation of TOAR DB [34] and ERA5 [35]. All four branches are then input to O3ResNet, which makes a four-day forecast of dma8 ozone.

CNN architecture Since we found the CNN architecture with residual blocks to be the best performing DL architecture on validation data, we describe the exact architecture in more detail below. This is the model we refer to as O3ResNet. The O3ResNet architecture consists of eight residual blocks, 20 hidden layers, and a total of about 800,000 trainable parameters. A residual block consists of two convolutional layers, where the first layer is bypassed by a skip connection to stabilize the training and thus allow training of deeper networks as gradients can propagate more directly during backpropagation [28]. We follow [36] and apply all convolutions only along the time axis. A special feature of the O3ResNet architecture are the four input branches, consisting of an LT and ST component of the chemical and meteorological inputs, respectively. The motivation for these separate branches is that the NN can initially learn local features of the different variable types, chemical and meteorological variables, and time scales, LT and ST, and later put this knowledge into a global context to make a prediction for ozone. The global context is learned in the tail of the network starting from a concatenation layer up to the output layer. Each branch consists of two convolutional layers with 32 7x1 and 32 3x1 filters and a maxpooling operation (with pool size 2x1), succeeded by four residual blocks with 32 3x1 filters and four residual blocks with 64 3x1 filters. The outputs of each branch are flattened and concatenated into a layer followed by a dense layer of 128 neurons and the output layer of four neurons, one for each day to be predicted. Except for the output layer, which features linear activation, all layers use a Parametric Rectified Linear Unit [PReLU, 37] activation function. The architecture of O3ResNet is shown in Figure 3. Further details on the O3ResNet's hyperparameters and on the alternative network architectures are given in Appendix B.

CAMS We compare O3ResNet against the state-of-the-art regional chemistry transport model ensemble with data assimilation from the Copernicus Atmosphere Monitoring Service (CAMS). The data are downloaded from the CAMS Atmosphere Data Store [39] and preserved on local systems as ADS hosts data on a rolling three-year archive. CAMS provides 96-h forecasts on a 0.1°x0.1° grid for Europe based on the median value approach of the nine ensemble members [40]. Details on the ensemble members are provided in Appendix C. To produce the CAMS ensemble forecast, the median is calculated for each pixel individually using interpolated forecasts of all ensemble members. As CAMS provides a grid forecast, we apply nearest neighbor interpolation to extract data at the station locations. We have also tested a bilinear interpolation as an alternative. Bilinear interpolation performed better at some stations and worse at others, so that on average the choice of interpolation method has no discernable effect on the CAMS performance. Finally, dma8 ozone is calculated from the hourly data at each station. [41] provide a detailed overview on CAMS.

Evaluation The final evaluation of the results is performed exclusively on the test data, which were neither used for training nor for hyperparameter optimization. For evaluation, we use the root mean squared error (RMSE) which is given by the square root of the MSE from Equation 1

$$\mathbf{RMSE} = \sqrt{\mathbf{MSE}} \tag{2}$$

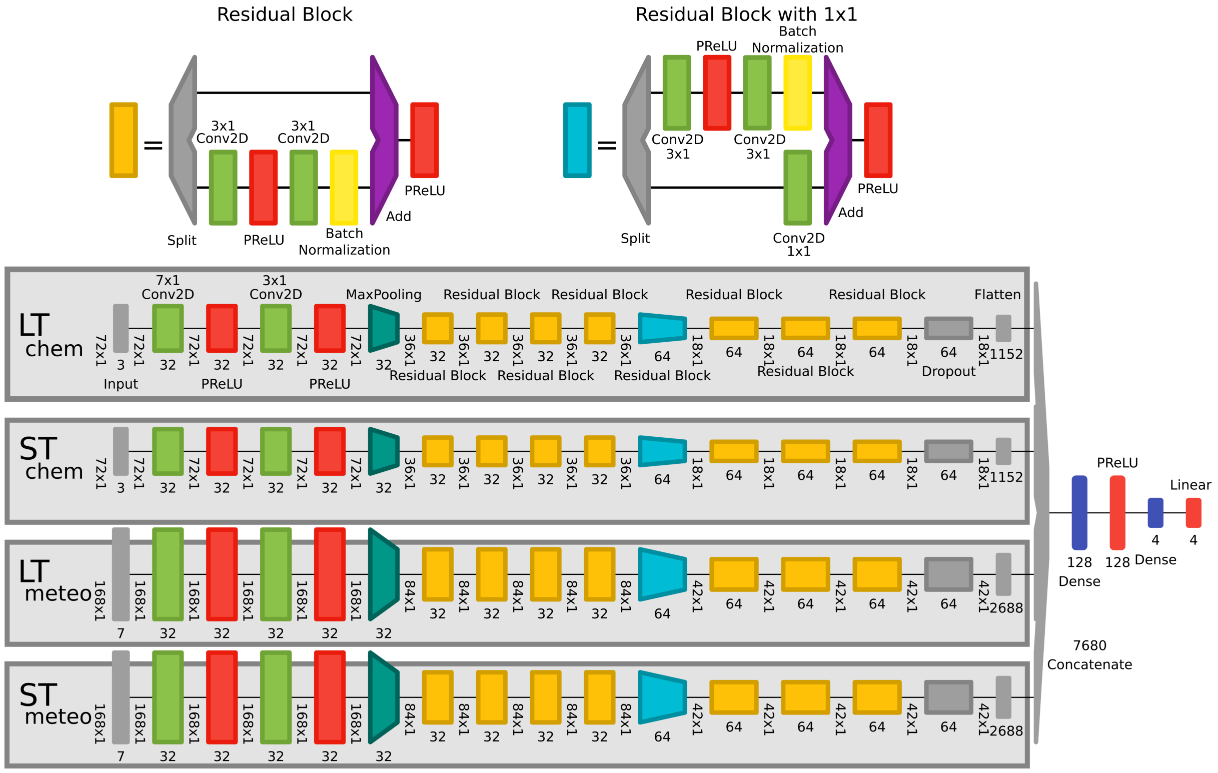


Figure 3: Network architecture of O3ResNet consisting of convolutional layers (green), PReLU and linear activation (red), maxpooling layers (teal), batch normalization layers (yellow), residual blocks (orange), residual blocks with additional 1x1 filter to increase number of filters (cyan), dense layers (blue), add layer (purple), and input, dropout, flatten, concatenate and split layers (all grey). Each branch is highlighted by a separate grey box. Numbers next to a layer show number of filters resp. weights and the shape. Shapes of the inputs correspond to 72 hourly values for three chemical variables on the interval $[t_0 - 3d, t_0]$ and to 168 hourly values for seven meteorological variables on $[t_0 - 3d, t_0 + 4d]$. The graphic is created with Net2Vis [38] and edited afterwards. A list of hyperparameters can be found in Table 5.

as well as the mean error (ME) given by

$$\mathbf{ME} = \frac{1}{n_i \cdot n_j} \sum_{i,j}^{n_i,n_j} \left(\hat{y}_{i,j} - y_{i,j}\right) = \bar{\hat{y}} - \bar{y} \tag{3}$$

which can also be expressed as the difference between the means of forecast $\bar{\hat{y}}$ and observation $\bar{y}$. To compare two models A and B against each other directly, we resort to the skill score given by

$$\mathbf{SS}\,(A, B) = 1 - \frac{\mathbf{MSE}_A}{\mathbf{MSE}_B}\ , \tag{4}$$

with $\mathbf{MSE}_A$ being the MSE of model A and $\mathbf{MSE}_B$ of model B.

3 Results

Figure 4 shows the RMSE as box-and-whiskers aggregated over all stations. O3ResNet yields a smaller RMSE for all forecast days compared to CAMS. O3ResNet achieves the smallest error for the D1 forecast with 4.3 ppb. The RMSE increases to 5.5 ppb for the D4 forecast, with almost identical RMSE on D3 and D4. Overall, the RMSE for O3ResNet lies between 3.9 ppb and 5.8 ppb regarding the 25th and 75th percentiles of all stations. CAMS, on the other hand, shows a noticeably higher RMSE, with a mean RMSE ranging from 7.3 ppb on D1 to 7.9 ppb on D4. Moreover, a wider spread of RMSE across stations can be seen for CAMS. Thus, the 25th and 75th percentiles are 6.5 ppb and

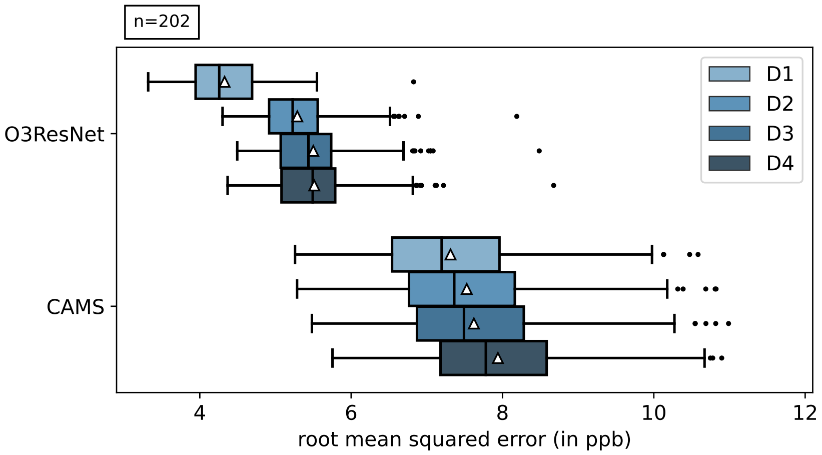


Figure 4: Distribution of the RMSE of O3ResNet and CAMS over all test stations visualized as box-and-whisker. The different shades of blue correspond to the error for D1 (light blue) to D4 (dark blue). The boxes indicate the 25th and 75th quantile of the distribution, the line within the box shows the median and the white triangle the mean.

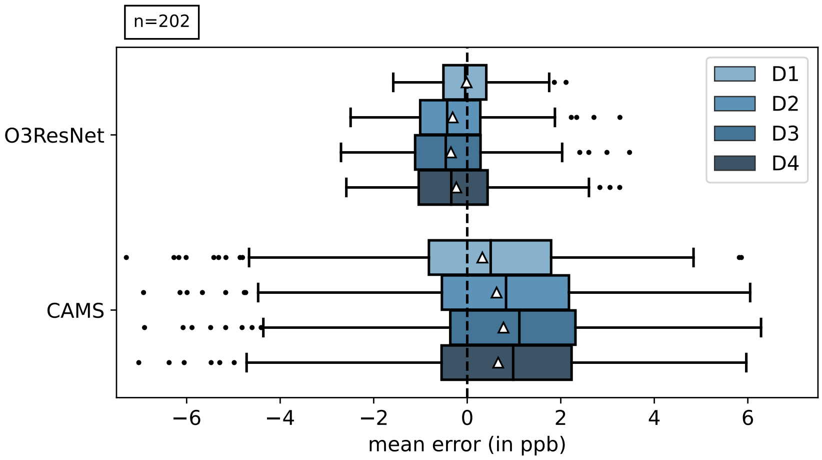


Figure 5: Similar to Figure 4, but here the ME is shown instead for O3ResNet and CAMS.

8.6 ppb, respectively. We also show the spatial distribution of the RMSE of O3ResNet and CAMS in the Appendix C (Figures 17 and 18).

The ME shown in Figure 5 provides insight into systematic biases between prediction and observation. For O3ResNet, we can see that the ME averaged over all stations is centered between -0.35 ppb and -0.01 ppb for all forecast days and with an interquartile range (IQR) between 0.92 ppb and 1.48 ppb. The ME for the CAMS predictions averages between $+0.32$ and $+0.78$ ppb, with the median for D2 to D4 being larger than $+0.83$ ppb. Overall, the CAMS ME shows a wide variation with an IQR of > 2.6 ppb.

The analysis of the ME shows that CAMS suffers from a consistent high bias in relation to the observations. Therefore, we next correct all forecasts of CAMS and O3ResNet by (1) removing the averaged background value for each station and (2) subtracting a 30-day running mean from the forecasts for each station. This reveals what contribution to the total error is due to an improper accounting of the variability of ozone and what contribution is due to a systematic deviation. Figure 6 shows the results for bias corrected predictions using method (1). Here, the adjustment leads to a reduction in the RMSE for the CAMS predictions. Accordingly, since O3ResNet exhibited already a low ME, this post-processing method does not lead to any improvement for O3ResNet. In contrast, the bias corrected forecasts using

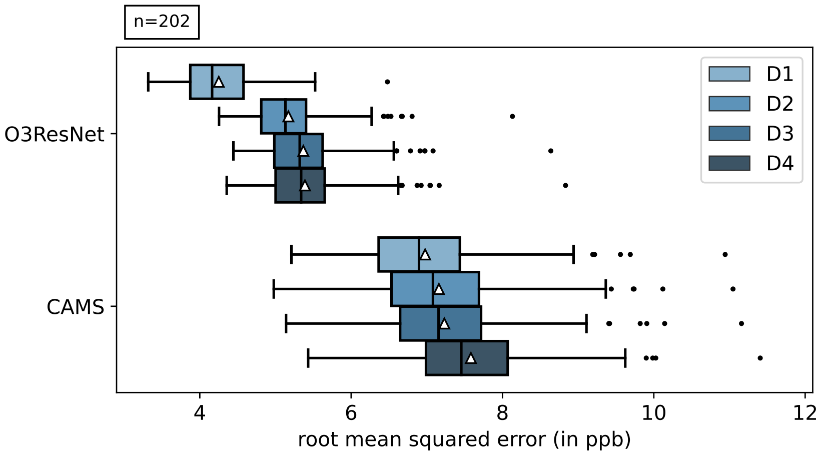


Figure 6: Similar to Figure 4, but here the bias-corrected RMSE is shown instead for O3ResNet and CAMS. Correction is applied by removing the average background concentration at each station.

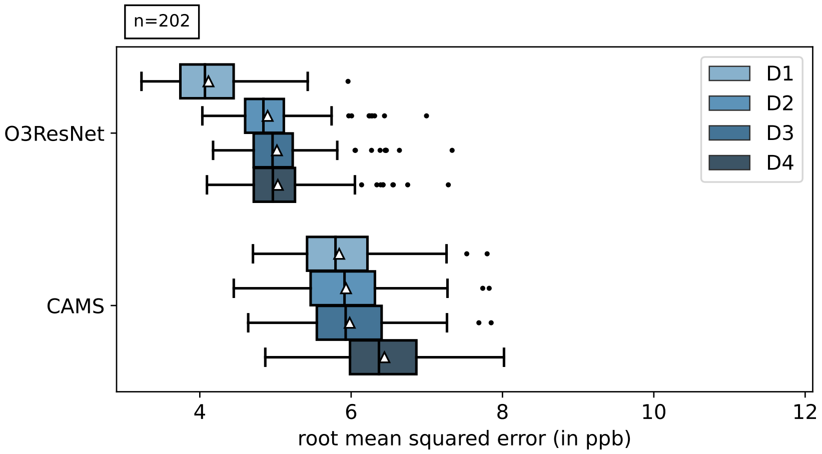


Figure 7: Similar to Figure 4, but here the seasonal bias-corrected RMSE is shown instead for O3ResNet and CAMS. For bias correction, we remove a 30-day running mean for each station.

method (2) lead to an improvement for CAMS and O3ResNet as measured by the RMSE (see Figure 7). In all cases, it can be concluded that O3ResNet can better represent the variability of ozone.

Since ozone concentrations exhibit pronounced seasonal variation and the variance also varies with season, we next consider the seasonality of the error. Figure 8 shows the RMSE aggregated over all forecast steps for each month across all stations for the entire test period. For each individual month, O3ResNet has a lower RMSE than CAMS. In addition, the IQR indicated by the width of the band of quantiles is narrower for O3ResNet. Both findings are in line with the results presented so far. Indeed, we can identify a season-dependent performance for both O3ResNet and CAMS in Figure 8. Overall, both models perform best in the spring months March, April, and May (MAM), whereas the summer months June, July, and August (JJA) show the highest error. It should be pointed out that O3ResNet can provide notably better forecasts than CAMS for JJA 2019, but for JJA 2021 neither model can provide decent forecasts, especially in July.

To provide further insight into the quality of the O3ResNet forecasts we use the likelihood-base rate factorization after [42]. Figure 9 compares observation and prediction of O3ResNet. Shown in the dashed lines is the conditional

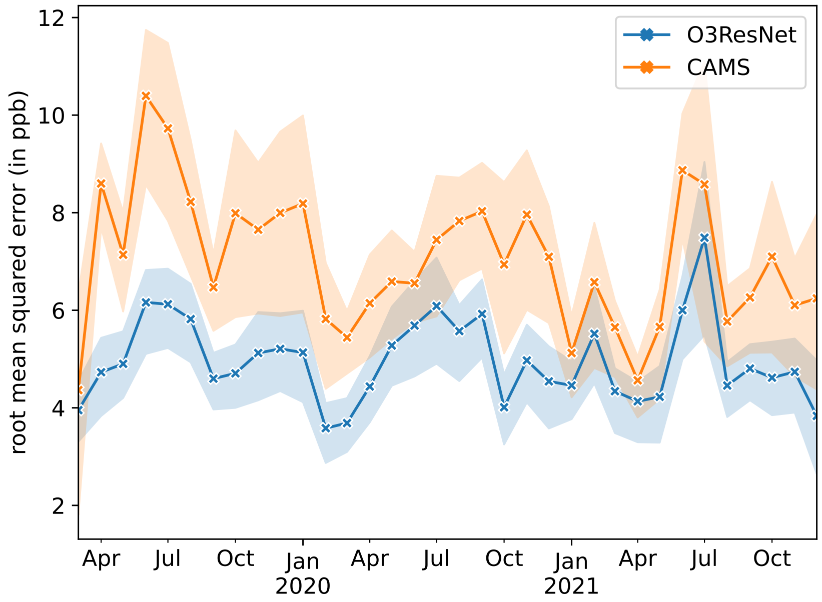


Figure 8: Month-to-month variation of performance (RMSE) for O3ResNet (blue) and CAMS (orange) during the test period. Mean RMSE over all stations is shown as thick line as well as crosses and 25th and 75th quantile are illustrated as bands.

distribution of the probability that, given a particular observation, O3ResNet can issue a proper forecast in advance. Considering the climatological distribution of the observations, represented by the gray bars (marginal distribution), this view allows to draw conclusions about how well O3ResNet can discriminate between different observation events [43]. It can be seen that the reference line and the median of the conditional quantile are in agreement in the range between 20 ppb and 55 ppb, and thus O3ResNet can distinguish individual observations well in this interval. However, for small ozone values, the model tends to overestimate slightly, indicated by the fold in the lines of the conditional quantiles. Also, for ozone values exceeding 60 ppb, 03ResNet cannot fully follow the observations, tending to underestimate the ozone concentration. However, observations of high ozone concentrations are severely underrepresented in the training data, and regression approaches such as O3ResNet generally tend to favour values towards the mean. Regarding the forecast horizon, increasing uncertainty with lead time is visible as the lines of the quantiles of the conditional distribution for D4 of the forecast are more widely spaced and both ends of the lines curve more pronouncedly than for D1. In the range from 20 ppb to 50 ppb, however, the reference lines and median continue to be close to each other indicating a reliable forecast issued by O3ResNet. The likelihood-base rate factorization for CAMS can be found in the Appendix C in Fig. 16. Here it can be seen that CAMS is not able to distinguish well between different observation events, because the slope of the conditional quantile lines deviates from the ideal reference line, meaning that smaller values are generally overestimated and high concentrations are underestimated.

3.1 Importance of input branches

To shed light on the robustness of the O3ResNet forecasts we follow the singlepass approach [44]. To understand the impact of each individual branch on O3ResNet, we fix all inputs of a single input branch to their average values and examine how much the resulting prediction differs from the unperturbed prediction. We measure this by the skill score as shown in Equation 4. The stronger the skill score of the mean-fixed O3ResNet decreases with respect to the original O3ResNet forecasts, the greater the influence of the respective branch. Results are presented in Figure 10. Considering all forecast days, the LT chemical and ST meteorological inputs have the strongest influence on the predicted ozone concentrations. The LT chemical inputs are particularly important for the D1 forecast, and appear to be less important for D2 to D4. Moreover, for the D1 forecast, the ST component of the chemical inputs is important to some extent, whereas for the other days this is not evident. LT meteorological inputs only play a minor role for O3ResNet for all forecast days. In contrast, the ST meteorological components are relevant for all forecast days and their importance even increases from D1 to the following days.

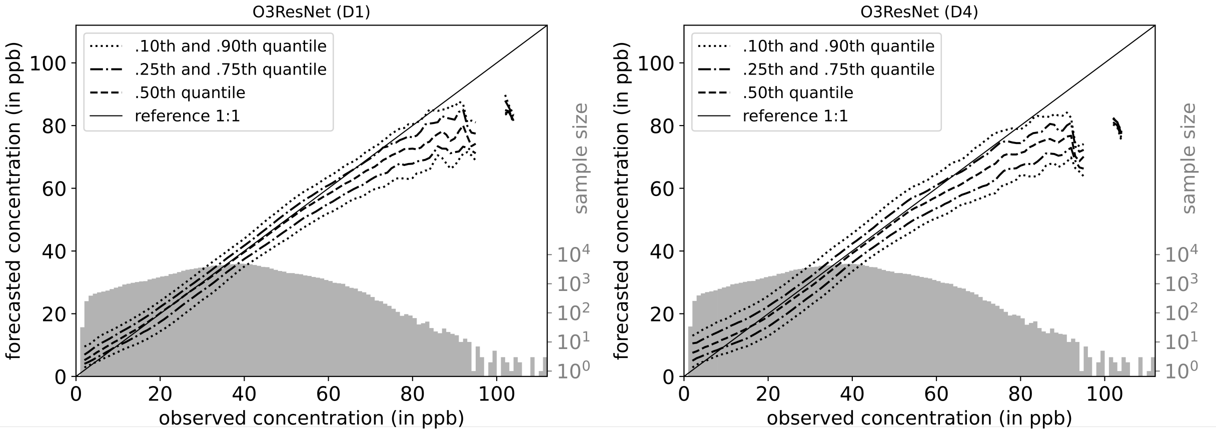


Figure 9: Visualization of the likelihood-base rate factorization for the D1 (left) and D4 (right) forecast of O3ResNet. The factorization consists of the conditional distribution of the probability that a prediction is made in advance of an incoming observation and the frequency distribution of the observations. The conditional distribution is represented by the 10th, 25th, 50th, 75th, and 90th quantiles using different dashed lines and the optimal reference line. The frequency distribution of the observations is shown by a histogram (gray bars) with logarithmic scale on the right axis.

From a meteorological perspective, these sensitivities can be explained as follows. The LT chemical inputs allow O3ResNet to perform a bias correction, as they provide information about the long-term background concentration. In addition, these components also add information about the season, since, for example, average ozone concentrations are higher in summer than in winter. Note that O3ResNet has no explicit information about the day or month of the samples it is processing. The relevance of the ST chemical variables can be explained by the autocorrelation of ozone. As it decreases with lead time, the importance of past observations also drops. By contrast, the LT components of the meteorological variables cannot add any valuable information to O3ResNet, since all information about seasonality is already contained in the LT chemical variables. However, the ST components of the meteorological inputs play an important role, since the deviations from long-term conditions contained therein characterize the current and future weather situation. For example, the ST meteorological variables provide information about the daily maximum of temperature and humidity in the forecast horizon.

3.2 Influence of the meteorological forecast lead time

Since this study uses ERA5 data as a pseudo-forecast and over an extended time horizon to calculate the LT and ST components (see Appendix A), questions arise on how O3ResNet would behave in an operational setting where meteorological forecasts have a more limited lead time and the forecast error tends to grow with increasing lead time. A sensitivity study, outlined subsequently, reveals that the forecast quality of O3ResNet is hardly affected by reducing the lead time of the meteorological forecast down to four days. To conduct this sensitivity study, we gradually decrease the maximum lead time for the meteorological variables. Values after this maximum lead time are replenished by the climatological statistics, as described in [32] and as it is done for the chemical variables. We do not retrain O3ResNet on these modified inputs, but analyze how O3ResNet responds to this new information and whether the skill of the ozone prediction decreases in dependence of the meteorological forecast lead time. Results are shown in Figure 11.

At large lead times it can be seen that the reduction of the lead time of the meteorological variables from 168 to 93 hours has no effect on the forecast performance of O3ResNet as the skill score stays close to zero indicating neither a gain nor loss of skill. Note, that we test with larger lead times than the four days forecast horizon of O3ResNet, as longer time series are mandatory to calculate an exact LT and ST decomposition (see Appendix A). As this analysis shows, a blurred decomposition does not decrease the model's performance at all. A further decrease of the lead time up to the extreme case of 0 hours results in a continuous decrease of the prediction skill for all days. Thereby, the forecast of O3ResNet always deteriorates only for the forecast days from which on no meteorological forecast is available and climatology is fallen back on as a substitute. For example, when the lead time of the meteorological variables is 48 hours, only the ozone forecasts for D3 and D4 worsen, with the D3 forecast having an equal skill to the CAMS forecast in this particular case. Conversely, the ozone forecasts for D1 and D2 are not affected at all and remain at their original skill level. This finding can be observed for all forecast days. Besides, results show that the D1 forecast of O3ResNet is more skillful than CAMS even at a lead time of 0 hours.

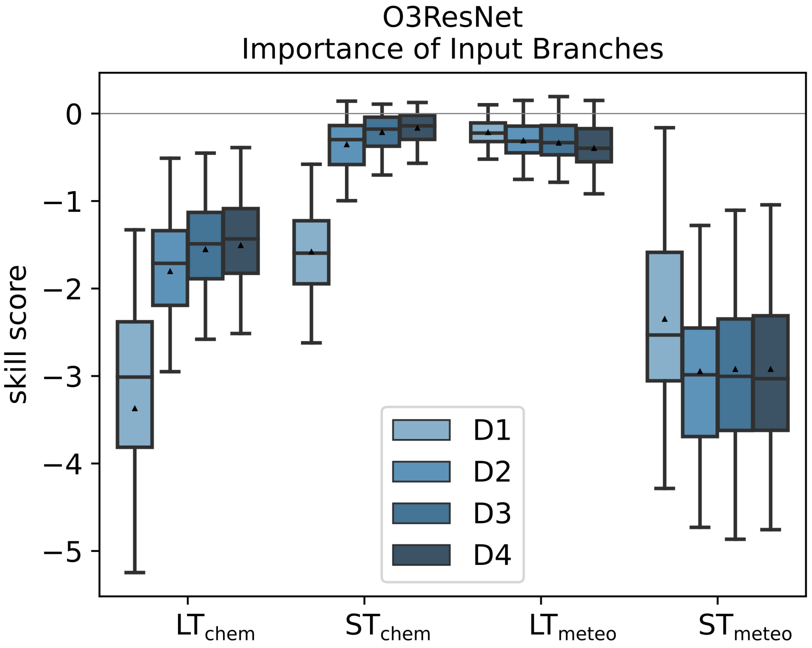


Figure 10: Evaluation of the importance of each input branch for the prediction of O3ResNet generated using the singlepass approach. The skill score is calculated in reference to the unperturbed prediction. The impact on each prediction day is shown by blue colors for D1 (light blue) to D4 (dark blue). A large negative value indicates a strong dependence whereas a value close to 0 describes a weaker dependence.

4 Discussion and Conclusions

This manuscript outlines the development of a skillful and reliable forecasting system for a 4-day point forecast of dma8 ozone based on DL methods. O3ResNet performs better than the state-of-the-art CAMS regional ensemble. O3ResNet was developed with data from Central Europe, but can easily be trained for other regions and, in principle, for other ozone metrics or even other air pollutants such as particulate matter or nitrogen oxides, provided sufficient data are available. The transferability of O3ResNet will be the subject of another study. The results above show that the combination of a CNN architecture with residual blocks, the temporal decomposition of inputs into long-term and short-term, and the integration of a weather forecast for all meteorological input parameters are the key ingredients for our new high-quality ozone forecasting system.

The outstanding advantages of O3ResNet are a nearly bias-free forecast as well as a low seasonal variation of the forecast quality. O3ResNet provides high quality predictions especially in the range of 20 to 55 ppb and for September to May. Only at the edges of the distribution and for forecasts during the summer season does the performance decrease a bit, although O3ResNet still outperforms the CAMS regional model ensemble. First, from a statistical point of view, this is related to heteroscedasticity, since the variability of ozone is very high in summer and lower in winter. Second, ozone in summer is more determined by the local daily maximum temperature [19], which is less well reflected in the meteorological forecasts due to limited spatial model resolution. While such processes generally pose a problem for conventional CTMs as well [15, 45], O3ResNet can at least better accommodate them. The nearly bias-free forecast can be attributed to O3ResNet's understanding of the LT chemical variables, which allows O3ResNet to determine a correct concentration level at the target station. The ST meteorological inputs have a major contribution to the O3ResNet forecast quality, because they provide information about the current weather situation.

Analysis of the dependence on the horizon of the weather forecast shows that O3ResNet can already provide a fully reliable forecast of future ozone concentrations with a weather forecast of similar lead time. With a 48-hour weather forecast, O3ResNet achieves an adequate 2-day forecast. This shows, with respect to previous studies such as [24] or [32], that ozone prediction with DL methods is limited not by understanding the relationship between weather and

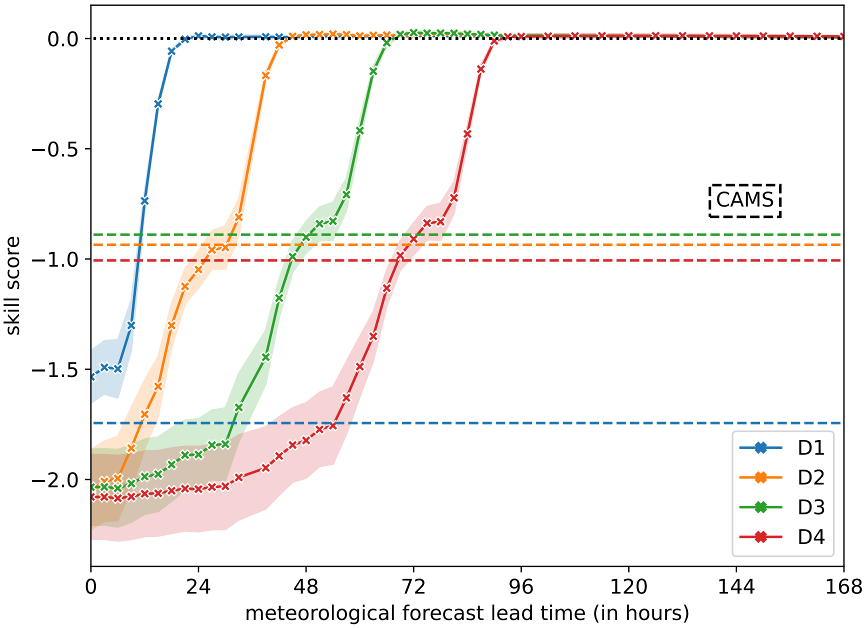


Figure 11: Skill score of the forecast quality of O3ResNet depending on the lead time of the meteorological forecast in relation to a forecast based with quasi unlimited lead time. The forecast days are individually colored for D1 (blue), D2 (orange), D3 (green) and D4 (red). The solid lines represent the mean skill scores, the bands the range between 25th and 75th quantile. Additionally, the skill scores for CAMS in relation to the original O3ResNet forecast are shown as dashed reference lines. Negative skill scores mean that the forecast for the corresponding meteorological forecast lead time is worse than the best case. At a skill score of zero, the difference disappears.

air quality but, in particular, by uncertainty about future weather, and that the inclusion of a skillful weather forecast contributes great value to DL-based ozone predictions.

In comparison to the CAMS regional ensemble median forecast, O3ResNet shows significant improvements for all forecast days. Moreover, CAMS requires additional post-processing to deliver forecasts on station level, whereas O3ResNet does not. [41] mention the development of various post-processing methods, including ML, to adapt the raw CAMS forecasts to point forecasts with higher skill that are expected to be deployed in the coming years. O3ResNet demonstrates that high-quality ozone forecasts do not necessarily require to run a complete CTM system, but can alternatively also be produced using DL plus weather forecasts, which is much faster. A four-day forecast at all 328 stations of this study takes about 10 seconds.

Concluding, we suggest a number of tests and improvements before applying O3ResNet operationally. First, ERA5 is no real forecast, but a reanalysis, meaning that the frequency of updates through data assimilation is much higher. Nevertheless, it can be reasonably expected that the forecast quality of O3ResNet would not drop dramatically, as relevant numerical weather prediction on comparable spatial and temporal resolution, such as the Integrated Forecast System (IFS) operated by the European Centre for Medium-Range Weather Forecasts (ECMWF), already provides a very reliable forecast for one week ahead [see 46, 47]. Second, O3ResNet is currently trained in rural and suburban areas on stations classified as background. To provide a full range of forecasts, the model should also be tested in urban areas as well as in regions with dominant air pollution sources, which may require the integration of emissions data. Third, it is recommended to further investigate the predictive power for peak ozone concentrations. Albeit O3ResNet is capable of simulating concentrations of dma8 ozone up to 80 ppb, the most extreme observed values are not reproduced satisfactorily. For example, O3ResNet for July 2021 does not match with observations well. Herein, uncertainty prediction, e.g., using probabilistic DL architectures as in [48] or following [49], who predict the parameters of a probability distribution instead of the deterministic values, could add useful information. Also, transformers [50] or more specifically a temporal fusion transformer [51], harbors promising potential. In combination with suitable interpolation techniques, such DL models may even be able to generate useful forecasts at locations where no measurements of air pollutant concentrations are performed.

Acknowledgments

We thank Dr. Amirpasha Mozaffari and Enxhi Krespha for downloading the CAMS data, CAMS and the TOAR data center teams for providing the data, and the Jülich Supercomputing Center for computing time under the DeepACF project. This research has been supported by the European Research Council, H2020 Research Infrastructures (IntelliAQ, grant 787576), and the DFG Funding Programme, Open Access Publication Funding (2022–2024).

Data availability statement

Input data, forecasts on test data, and O3ResNet model are openly available from http://doi.org/10.34730/76529959732a464486ec5b9277152233

APPENDIX A

Cross-validation

We perform cross-validation of the best model architecture (O3ResNet) by rotating the subsets, keeping the length of each subset, 3 years for validation and testing and 12 years for training, as well as the hyperparameter configuration. Data are always split block-wise along time. Therefore, in total, we test six different arrangements. Results are shown in Table 1 and Figure 12. It can be seen, that the RMSE is close for all orderings of subsets. Yet, there is a deviation in performance when positioning testing phase at the very beginning. Note that the number of samples varies from about 160,000 (train/val/test) to 225,000 (test/train/val) due to a large temporal and spatial variability of data coverage. In Figure 13, we furthermore show the temporal distribution of the target dma8 ozone in the final subset ordering (train/val/test). It can be seen that the temporal distribution is quite similar for all subsets.

Table 1: Tabular results of cross-validation implemented by rotating training, validation and testing subsets. The RMSE is shown in ppb and also visualized in Figure 12.

RMSE	D1	D2	D3	D4	mean
data split					D1-D4
train/val/test	4.55	5.43	5.69	5.73	5.35
train/test/val	4.59	5.56	5.64	5.72	5.38
val/train/test	4.25	5.14	5.25	5.38	5.01
val/test/train	4.71	5.71	5.87	5.90	5.55
test/train/val	5.13	6.28	6.45	6.51	6.09
test/val/train	5.25	6.42	6.69	6.73	6.27

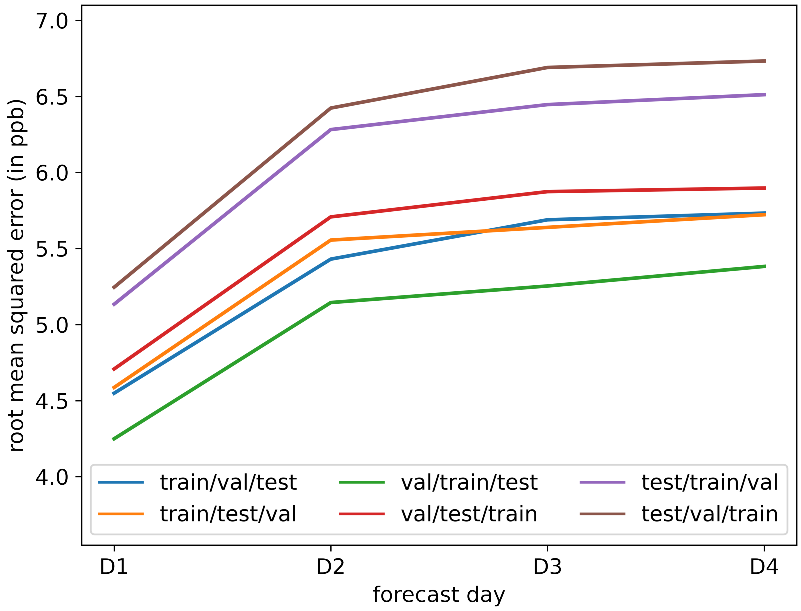


Figure 12: Visualization of cross-validation results as shown in Table 1.

Filtering of data

All time series are split into long-term (LT) and short-term (ST) components by means of FIR filter with a Kaiser window [52] with parameter $\beta = 5$, a cutoff-period of 21 days and order of $N = 42$ days. For applying the FIR filter causally to all chemical variables, we follow the approach from [32] and use climatology for time steps in the lead time, whereas reanalysis data are used as a pseudo-forecast for the meteorological variables.

The decomposition is formalized by the following steps. First, we calculate a climatological statistic a_i which contains the seasonal cycle of the monthly mean as well as the diurnal cycle. Heteroscedasticity is taken into account by allowing this diurnal cycle to vary over the year.

$$a_i = f\big(x_i,\ t_i\big) \tag{5}$$

A composite time series $\breve{x}_i$ is created from the raw time series x_i and the climatological statistic a_i for each time t_0 at which a forecast is initiated. The combination is done depending on the lead time t_l. For the chemical variables, $t_l = 0$ always applies, and for the meteorological variables, $t_l \to \infty$. For the analysis of the dependence of O3ResNet on the lead time of the meteorological variables, t_l is set to a lead time between 0 and 168 hours accordingly.

$$\breve{x}_i(t_0) = \begin{cases} x_i & , t_i \leq t_0 + t_l \\ a_i & , t_i > t_0 + t_l \end{cases} \tag{6}$$

The properties b_i of the FIR filter are determined by the Kaiser window given for the order of $N = 42$ days. Applying the filter results in the LT components $x_i^{(LT)}$ of the time series.

$$x_n^{(LT)}(t_0) = \sum_{i=t_0-N/2}^{t_0+N/2} b_i \cdot \breve{x}_{n-i}(t_0) \tag{7}$$

Finally, the ST components $x_i^{(ST)}$ are calculated by the difference between the original time series x_i and the LT components $x^{(LT)}$.

$$x_i^{(ST)}(t_0) = x_i - x_i^{(LT)}(t_0) \tag{8}$$

This means in reverse that the sum of LT and ST components always adds up to the original time series.

Temporal distribution of dma8 ozone

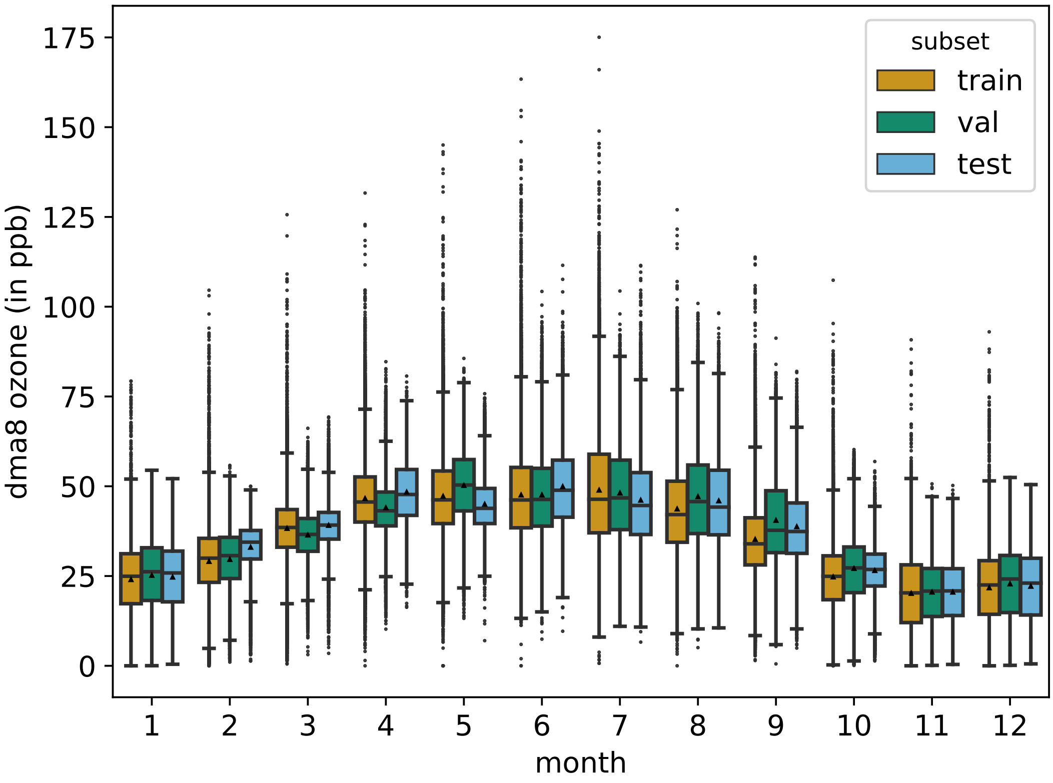


Figure 13: Temporal distribution of dma8 ozone aggregated over all observations and stations illustrated as box-and-whiskers. Distribution of the training (orange), validation (green) and testing (blue) data are highlighted in color.

APPENDIX B

Technical details

We train all NNs for this study on the Helmholtz Data Federation Machine Learning System (HDF-ML) at the Jülich Supercomputing Centre in Jülich, Germany. In total, HDF-ML is equipped with 15 compute nodes, each running 4 Nvidia Tesla V100 GPUs and 2 Intel Xeon Gold 6126 with 12 cores (24HT). For each training, we use a single node with all available GPUs since the computation times of the training are moderate (between half an hour and up to four hours). Training as well as pre- and post-processing are carried out with the research software MLAir [53]. MLAir is based on the programming language python, provides a complete workflow for performing ML experiments with a special focus on time series predictions [54] and thereby makes use of tensorflow [55] for the ML training.

Preprocessing of the raw data of a single station covering the entire time period takes on average 108 sec, which means that preprocessing of a single sample is about 0.03 sec on average. Approximately 90 % of the preprocessing time is spent calculating the decomposition into LT and ST components, as the data for each sample changes with t_0. For this study, we use 12 parallel threads, so preprocessing is 12x faster on our systems. The inference time for a single station is approximately 2.8 sec (0.0009 sec per sample). Measured inference time includes losses due to I/O operations such as loading the processed data from disk and storing the predictions locally. The actual NN prediction, without I/O operations, is performed on 4 GPUs in parallel. Numbers are also shown in Table 2.

Table 2: Preprocessing and inference time.

operation	data	duration (in sec)
preprocessing	station	108
preprocessing	sample	0.03
inference	station	2.8
inference	sample	0.0009

Hyperparameter tuning strategy

We apply a kind of evolutionary algorithm when searching for optimal hyperparameters. For the initial first generation, we randomly draw 70 combinations of hyperparameters according to the range of values, the sampling mode, and the variation properties shown in Table 3 and 4 and measure the validation error. For the second generation, we select the top 10 performing hyperparameter combinations in terms of validation error and again draw random combinations from this new set, allowing all parameters to further vary according to the specified variation properties. We do not test the exact same combination a second time. For the second generation, we reduce the number of experiments by 30%. In each subsequent generation, we apply the same scheme, but reduce the number of best performing combinations by 1 and the number of experiments by 30% each time. After running 10 generations, we consider the combination that leads to the lowest validation error across generations as the optimal choice of hyperparameters. We apply this search strategy separately for each NN architecture.

Table 3: Overview on all hyperparameters tuned in this study. Each parameter is selected from the given range and with indicated sampling method. Moreover, continuous parameter are varied according to the variation ratio. Details on the NN architectures are provided in Table 4. Parameters marked with † are not tested for ResNet and U-Net.

parameter	range	sampling	variation
learning rate	$[0.0001, \dots, 0.1]$	power of ten	80%
learning rate decay	$[0, 0.001, \dots, 0.1]$	power of ten	50%
batch size	$\{256,\ 512,\ 1024\}$	discrete	-
dropout	$[0, \dots, 0.7]$	linear	50%
batch normalization	$\{\text{true}, \text{false}\}$	discrete	-
l1 regularizer	$[0, 0.001, \dots, 0.1]$	power of ten	50%
l2 regularizer	$[0, 0.001, \dots, 0.1]$	power of ten	50%
activation function	$\left\{\text{relu}, \text{leakyrelu}, \text{prelu}, \text{elu}^{\dagger}, \text{selu}^{\dagger}, \text{tanh}^{\dagger}\right\}$	discrete	-
NN architecture	see Table 4	discrete	-

Table 4: List of NN specific hyperparameters referring to the model architecture. The model column contains information about the chosen architecture and number of different configurations. A slash in the values column indicates the number of neurons respective filters per layer.

model	parameter	values
FCN (10x)		
	hidden layers and neurons	$\{32, 64, 64/32, 128/32, 128/64, 128/64/32\ \ldots, 512/256/128\}$
	dense layer (after concat)	$\{\text{no}, 256, 256/64, 256/64/16\}$
CNN (12x)		
	layers	$[1, 2,\ \ldots, 6]$
	max pooling	{no, every 2nd layer, always}
	kernel size	$\{(3, 1), (5, 1)\}$
	filter	$\{16, 32, 64, 128\}$
	dense layer (after concat)	$\{\text{no}, 128, 256\}$
RNN (10x)		
	recurrent layer	$\{10, 32, 32/32, 64, 64/64, 64/32,\ \ldots, 256/128\}$
	unit type	{LSTM, GRU}
	dense layer (after concat)	$\{\text{no}, 32, 64, 128\}$
ResNet (16x)		
	residual blocks	$[6, 7,\ \ldots, 12]$
	kernel size	$\{(3, 1)\}$
	filter	$\{16/32/64, 16/32, 32/64, 32/64/128\}$
	consecutive layers with same filter	$[2, 4]$
	dense layer (after concat)	$\{\text{no}, 128\}$
U-Net (9x)		
	down blocks with filter	$\{16/32, 16/32/64, 16/32/64/128\}$
	kernel size	$\{(3, 1)\}$
	dense layer (before concat)	$\{\text{no}, 128\}$
	dense layer (after concat)	$\{\text{no}, 128\}$
all		
	dropout	{no, only final layer, every 2nd layer, always}
	output activation	{linear}

Model selection

To select the best DL architecture, we look at the RMSE over all stations (Figure 14). It can be seen that the forecasts of the CNN with residual blocks (ResNet) and U-Net are with an average RMSE of 5.1 ppb better than those of the other DL models (between 5.6 and 5.8 ppb). However, the distributions of the RMSE for ResNet and U-Net do not differ significantly in a Mann-Whitney U test. Therefore, we apply a bootstrap procedure with 1000 repetitions as a second evaluation step. We split the entire test data set into monthly blocks, randomly sample 36 blocks with replacement for each iteration, and calculate the RMSE on each sample. In the bootstrap approach as shown in Figure 15, the ResNet architecture performs slightly better, so we use it for further analysis.

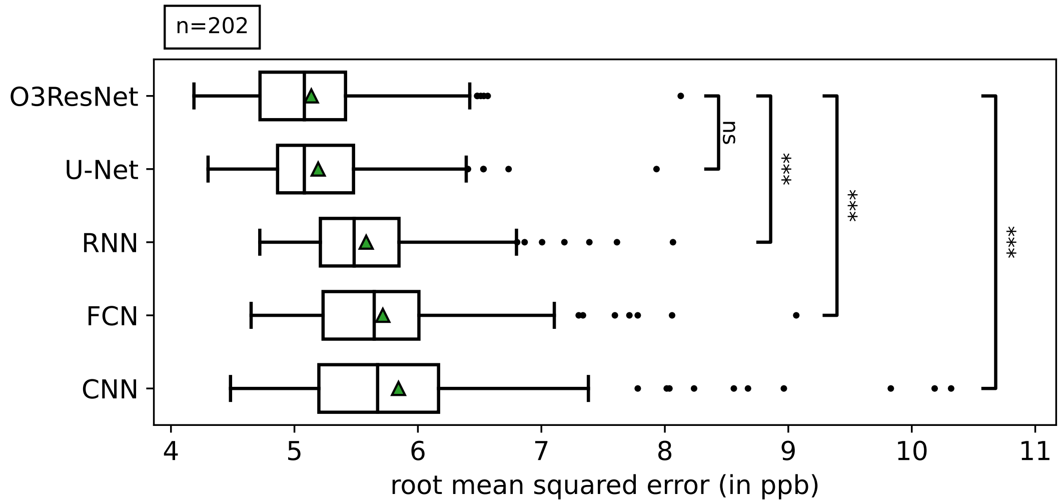


Figure 14: Distribution of the RMSE aggregated over test data ($n = 202$ stations) visualized as box-and-whiskers. Results from a Mann-Whitney U test are shown additionally. Three stars indicate a significance level of $p < 0.001$ and "ns" (not significant) corresponds to $p > 0.05$.

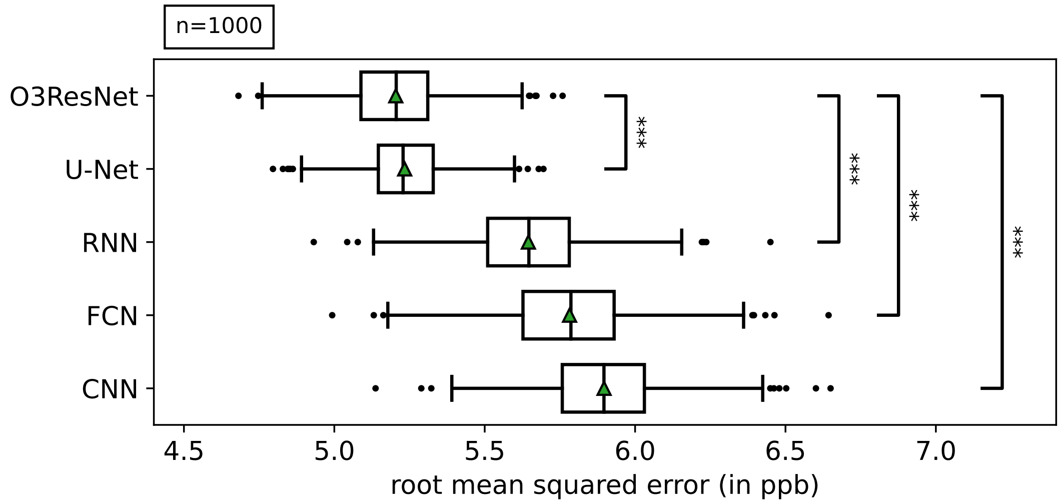


Figure 15: Distribution of the RMSE calculated on $n = 1000$ bootstrap samples (with replacement) plotted as box-and-whiskers. Significance levels are same as in Figure 14.

Table 5: Summary of the hyperparameters of O3ResNet.

parameter	range
learning rate	0.0003
learning rate decay	0.0
batch size	1024
dropout	0.59
batch normalization	false
l1 regularizer	0.095
l2 regularizer	0.12
activation function	prelu
NN architecture	see Figure 3
trainable parameters	807,812

APPENDIX C

Additional information on CAMS

A good overview of the regional CAMS ensemble can be found in [41]. The regional CAMS ensemble is composed of the nine members CHIMERE [56], DEHM [57], EMEP [58], EURAD-IM [59, 60], GEM-AQ [61], LOTOS-EUROS [62], MATCH [63, 64], MOCAGE [65, 66] and SILAM [67]. Each model is first interpolated on a 0.1°x0.1° grid individually and then the median is calculated for each grid cell. More information about this median value approach and in-depth details about the ensemble members involved are presented in [40].

Joint distribution of CAMS

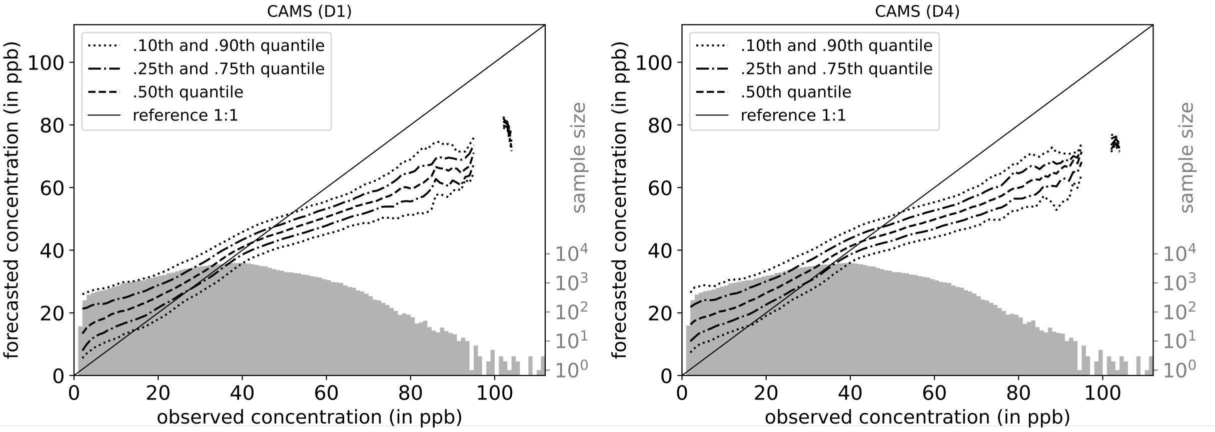


Figure 16: Visualization of the likelihood-base rate factorization for the D1 (left) and D4 (right) forecast of CAMS as in Fig. 9

Error maps

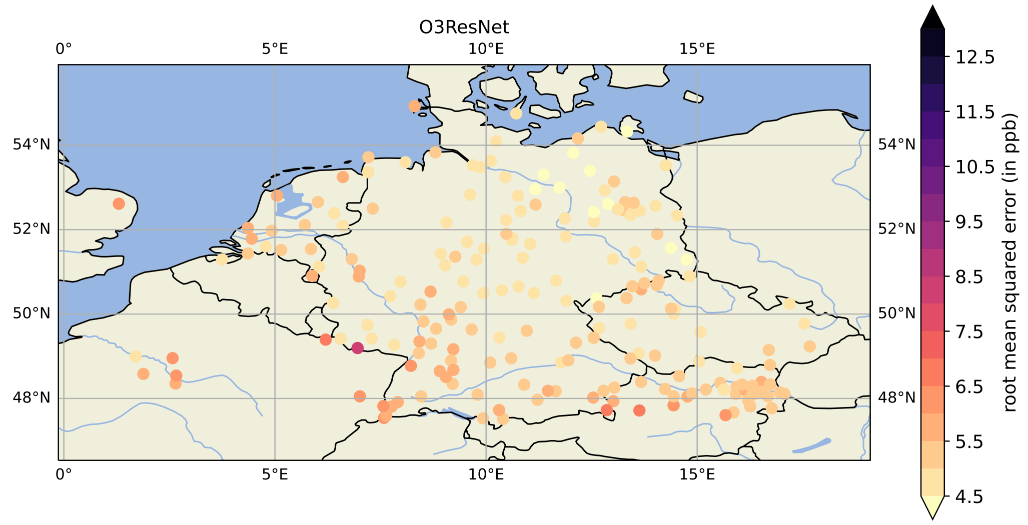


Figure 17: Spatial distribution of the RMSE of O3ResNet averaged on all forecast days at each observation station.

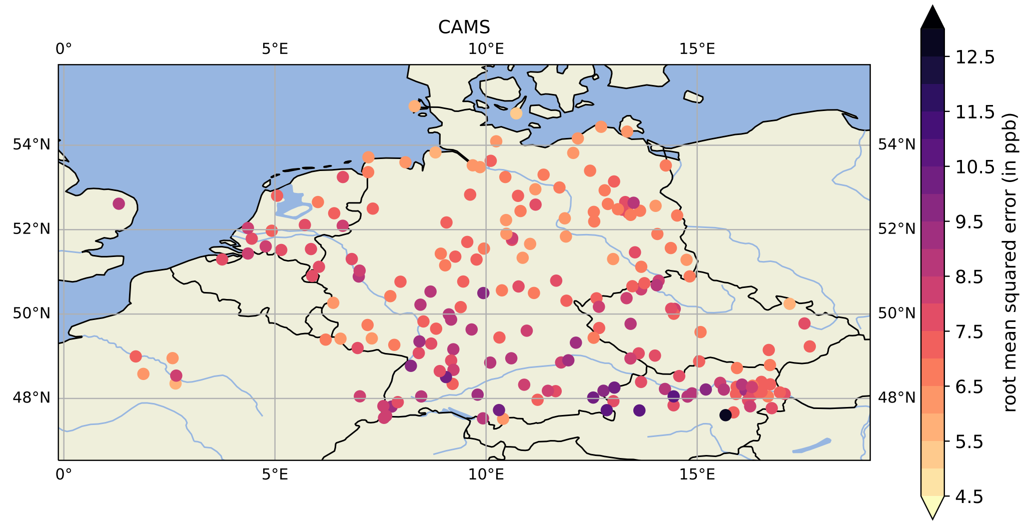


Figure 18: Same as in Figure 17 but for CAMS forecast.

References

[1] M. G. Schultz, C. Betancourt, B. Gong, F. Kleinert, M. Langguth, L. H. Leufen, A. Mozaffari, and S. Stadtler. Can deep learning beat numerical weather prediction? *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 379(2194):20200097, April 2021. ISSN 1364-503X, 1471-2962. doi:10.1098/rsta.2020.0097. URL `https://royalsocietypublishing.org/doi/10.1098/rsta.2020.0097`.

[2] Sverre Solberg, Augustin Colette, and Cristina Guerreiro. Discounting the impact of meteorology to the ozone concentration trends. *European Topic Centre on Air Pollution and Climate Change Mitigation, Bilthoven, the Netherlands, Technical Paper*, 9(2015/09), 2016.

[3] US EPA. Final report: Integrated science assessment of ozone and related photochemical oxidants. *US Environmental Protection Agency, Washington, DC*, 2013.

[4] P. S. Monks, A. T. Archibald, A. Colette, O. Cooper, M. Coyle, R. Derwent, D. Fowler, C. Granier, K. S. Law, G. E. Mills, D. S. Stevenson, O. Tarasova, V. Thouret, E. von Schneidemesser, R. Sommariva, O. Wild, and M. L. Williams. Tropospheric ozone and its precursors from the urban to the global scale from air quality to short-lived climate forcer. *Atmospheric Chemistry and Physics*, 15(15):8889–8973, August 2015. ISSN 1680-7324. doi:10.5194/acp-15-8889-2015. URL `https://acp.copernicus.org/articles/15/8889/2015/`.

[5] Gina Mills, Håkan Pleijel, Christopher S. Malley, Baerbel Sinha, Owen R. Cooper, Martin G. Schultz, Howard S. Neufeld, David Simpson, Katrina Sharps, Zhaozhong Feng, Giacomo Gerosa, Harry Harmens, Kazuhiko Kobayashi, Pallavi Saxena, Elena Paoletti, Vinayak Sinha, and Xiaobin Xu. Tropospheric Ozone Assessment Report: Present-day tropospheric ozone distribution and trends relevant to vegetation. *Elementa: Science of the Anthropocene*, 6:47, January 2018. ISSN 2325-1026. doi:10.1525/elementa.302. URL `https://online.ucpress.edu/elementa/article/doi/10.1525/elementa.302/112843/Tropospheric-Ozone-Assessment-Report-Present-day`.

[6] Zoë L. Fleming, Ruth M. Doherty, Erika von Schneidemesser, Christopher S. Malley, Owen R. Cooper, Joseph P. Pinto, Augustin Colette, Xiaobin Xu, David Simpson, Martin G. Schultz, Allen S. Lefohn, Samera Hamad, Raeesa Moolla, Sverre Solberg, and Zhaozhong Feng. Tropospheric Ozone Assessment Report: Present-day ozone distribution and trends relevant to human health. *Elementa: Science of the Anthropocene*, 6:12, January 2018. ISSN 2325-1026. doi:10.1525/elementa.273. URL `https://online.ucpress.edu/elementa/article/doi/10.1525/elementa.273/112792/Tropospheric-Ozone-Assessment-Report-Present-day`.

[7] WHO. Review of evidence on health aspects of air pollution: REVIHAAP project: technical report. Technical documents, World Health Organization. Regional Office for Europe, 2013.

[8] Michelle L. Bell, Antonella Zanobetti, and Francesca Dominici. Who is More Affected by Ozone Pollution? A Systematic Review and Meta-Analysis. *American Journal of Epidemiology*, 180(1):15–28, July 2014. ISSN 0002-9262, 1476-6256. doi:10.1093/aje/kwu115. URL `https://academic.oup.com/aje/article-lookup/doi/10.1093/aje/kwu115`.

[9] US EPA. Integrated Science Assessment (ISA) for Ozone and Related Photochemical Oxidants (Final Report, April 2020), 2020.

[10] A. M. M. Manders, E. van Meijgaard, A. C. Mues, R. Kranenburg, L. H. van Ulft, and M. Schaap. The impact of differences in large-scale circulation output from climate models on the regional modeling of ozone and PM. *Atmospheric Chemistry and Physics*, 12(20):9441–9458, October 2012. ISSN 1680-7324. doi:10.5194/acp-12-9441-2012. URL `https://acp.copernicus.org/articles/12/9441/2012/`.

[11] Robert Vautard, Michael D. Moran, Efisio Solazzo, Robert C. Gilliam, Volker Matthias, Roberto Bianconi, Charles Chemel, Joana Ferreira, Beate Geyer, Ayoe B. Hansen, Amela Jericevic, Marje Prank, Arjo Segers, Jeremy D. Silver, Johannes Werhahn, Ralf Wolke, S.T. Rao, and Stefano Galmarini. Evaluation of the meteorological forcing used for the Air Quality Model Evaluation International Initiative (AQMEII) air quality simulations. *Atmospheric Environment*, 53:15–37, June 2012. ISSN 13522310. doi:10.1016/j.atmosenv.2011.10.065. URL `https://linkinghub.elsevier.com/retrieve/pii/S1352231011011605`.

[12] Dominik Brunner, Nicholas Savage, Oriol Jorba, Brian Eder, Lea Giordano, Alba Badia, Alessandra Balzarini, Rocío Baró, Roberto Bianconi, Charles Chemel, Gabriele Curci, Renate Forkel, Pedro Jiménez-Guerrero, Marcus Hirtl, Alma Hodzic, Luka Honzak, Ulas Im, Christoph Knote, Paul Makar, Astrid Manders-Groot, Erik van Meijgaard, Lucy Neal, Juan L. Pérez, Guido Pirovano, Roberto San Jose, Wolfram Schröder, Ranjeet S. Sokhi, Dimiter Syrakov, Alfreida Torian, Paolo Tuccella, Johannes Werhahn, Ralf Wolke, Khairunnisa Yahya, Rahela Zabkar, Yang Zhang, Christian Hogrefe, and Stefano Galmarini. Comparative analysis of meteorological performance of coupled chemistry-meteorology models in the context of AQMEII phase 2. *Atmo-*

spheric Environment, 115:470–498, August 2015. ISSN 13522310. doi:10.1016/j.atmosenv.2014.12.032. URL https://linkinghub.elsevier.com/retrieve/pii/S1352231014009807.

[13] Bertrand Bessagnet, Guido Pirovano, Mihaela Mircea, Cornelius Cuvelier, Armin Aulinger, Giuseppe Calori, Giancarlo Ciarelli, Astrid Manders, Rainer Stern, Svetlana Tsyro, Marta García Vivanco, Philippe Thunis, Maria-Teresa Pay, Augustin Colette, Florian Couvidat, Frédérik Meleux, Laurence Rouïl, Anthony Ung, Sebnem Aksoyoglu, José María Baldasano, Johannes Bieser, Gino Briganti, Andrea Cappelletti, Massimo D'Isidoro, Sandro Finardi, Richard Kranenburg, Camillo Silibello, Claudio Carnevale, Wenche Aas, Jean-Charles Dupont, Hilde Fagerli, Lucia Gonzalez, Laurent Menut, André S. H. Prévôt, Pete Roberts, and Les White. Presentation of the EURODELTA III intercomparison exercise – evaluation ofthe chemistry transport models' performance on criteria pollutants and jointanalysis with meteorology. *Atmospheric Chemistry and Physics*, 16(19):12667–12701, October 2016. ISSN 1680-7324. doi:10.5194/acp-16-12667-2016. URL https://acp.copernicus.org/articles/16/12667/2016/.

[14] N. Otero, J. Sillmann, K. A. Mar, H. W. Rust, S. Solberg, C. Andersson, M. Engardt, R. Bergström, B. Bessagnet, A. Colette, F. Couvidat, C. Cuvelier, S. Tsyro, H. Fagerli, M. Schaap, A. Manders, M. Mircea, G. Briganti, A. Cappelletti, M. Adani, M. D'Isidoro, M.-T. Pay, M. Theobald, M. G. Vivanco, P. Wind, N. Ojha, V. Raffort, and T. Butler. A multi-model comparison of meteorological drivers of surface ozone over Europe. *Atmospheric Chemistry and Physics*, 18(16):12269–12288, August 2018. ISSN 1680-7324. doi:10.5194/acp-18-12269-2018. URL https://acp.copernicus.org/articles/18/12269/2018/.

[15] Z. S. Stock, M. R. Russo, and J. A. Pyle. Representing ozone extremes in European megacities: the importance of resolution in a global chemistry climate model. *Atmospheric Chemistry and Physics*, 14(8):3899–3912, April 2014. ISSN 1680-7324. doi:10.5194/acp-14-3899-2014. URL https://acp.copernicus.org/articles/14/3899/2014/.

[16] S. W. Wang, H. Levy, G. Li, and H. Rabitz. Fully equivalent operational models for atmospheric chemical kinetics within global chemistry-transport models. *Journal of Geophysical Research: Atmospheres*, 104(D23): 30417–30426, December 1999. ISSN 01480227. doi:10.1029/1999JD900830. URL http://doi.wiley.com/10.1029/1999JD900830.

[17] A. Baklanov, K. Schlünzen, P. Suppan, J. Baldasano, D. Brunner, S. Aksoyoglu, G. Carmichael, J. Douros, J. Flemming, R. Forkel, S. Galmarini, M. Gauss, G. Grell, M. Hirtl, S. Joffre, O. Jorba, E. Kaas, M. Kaasik, G. Kallos, X. Kong, U. Korsholm, A. Kurganskiy, J. Kushta, U. Lohmann, A. Mahura, A. Manders-Groot, A. Maurizi, N. Moussiopoulos, S. T. Rao, N. Savage, C. Seigneur, R. S. Sokhi, E. Solazzo, S. Solomos, B. Sørensen, G. Tsegas, E. Vignati, B. Vogel, and Y. Zhang. Online coupled regional meteorology chemistry models in Europe: current status and prospects. *Atmospheric Chemistry and Physics*, 14(1):317–398, January 2014. ISSN 1680-7324. doi:10.5194/acp-14-317-2014. URL https://acp.copernicus.org/articles/14/317/2014/.

[18] Arlene M. Fiore, Vaishali Naik, and Eric M. Leibensperger. Air Quality and Climate Connections. *Journal of the Air & Waste Management Association*, 65(6):645–685, June 2015. ISSN 1096-2247, 2162-2906. doi:10.1080/10962247.2015.1040526. URL https://www.tandfonline.com/doi/full/10.1080/10962247.2015.1040526.

[19] N Otero, J Sillmann, J L Schnell, H W Rust, and T Butler. Synoptic and meteorological drivers of extreme ozone concentrations over Europe. *Environmental Research Letters*, 11(2):024005, February 2016. ISSN 1748-9326. doi:10.1088/1748-9326/11/2/024005. URL https://iopscience.iop.org/article/10.1088/1748-9326/11/2/024005.

[20] Sally Jahn and Elke Hertig. Modeling and projecting health-relevant combined ozone and temperature events in present and future Central European climate. *Air Quality, Atmosphere & Health*, 14(4):563–580, April 2021. ISSN 1873-9318, 1873-9326. doi:10.1007/s11869-020-00961-0. URL https://link.springer.com/10.1007/s11869-020-00961-0.

[21] Xiang Weng, Grant L. Forster, and Peer Nowack. A machine learning approach to quantify meteorological drivers of ozone pollution in China from 2015 to 2019. *Atmospheric Chemistry and Physics*, 22(12):8385–8402, June 2022. ISSN 1680-7324. doi:10.5194/acp-22-8385-2022. URL https://acp.copernicus.org/articles/22/8385/2022/.

[22] Karl M. Seltzer, Drew T. Shindell, Prasad Kasibhatla, and Christopher S. Malley. Magnitude, trends, and impacts of ambient long-term ozone exposure in the United States from 2000 to 2015. *Atmospheric Chemistry and Physics*, 20(3):1757–1775, February 2020. ISSN 1680-7324. doi:10.5194/acp-20-1757-2020. URL https://acp.copernicus.org/articles/20/1757/2020/.

[23] Alqamah Sayeed, Yunsoo Choi, Ebrahim Eslami, Yannic Lops, Anirban Roy, and Jia Jung. Using a deep convolutional neural network to predict 2017 ozone concentrations, 24 hours in advance. *Neural Networks*,

121:396–408, January 2020. ISSN 08936080. doi:10.1016/j.neunet.2019.09.033. URL https://linkinghub.elsevier.com/retrieve/pii/S0893608019303156.

[24] Felix Kleinert, Lukas H. Leufen, and Martin G. Schultz. IntelliO3-ts v1.0: a neural network approach to predict near-surface ozone concentrations in Germany. *Geoscientific Model Development*, 14(1):1–25, January 2021. ISSN 1991-9603. doi:10.5194/gmd-14-1-2021. URL https://gmd.copernicus.org/articles/14/1/2021/.

[25] Jun Ma, Zheng Li, Jack C.P. Cheng, Yuexiong Ding, Changqing Lin, and Zherui Xu. Air quality prediction at new stations using spatially transferred bi-directional long short-term memory network. *Science of The Total Environment*, 705:135771, February 2020. ISSN 00489697. doi:10.1016/j.scitotenv.2019.135771. URL https://linkinghub.elsevier.com/retrieve/pii/S0048969719357663.

[26] Tai-Long He, Dylan B. A. Jones, Kazuyuki Miyazaki, Binxuan Huang, Yuyang Liu, Zhe Jiang, E. Charlie White, Helen M. Worden, and John R. Worden. Deep learning to evaluate US NO $_x$ emissions using surface ozone predictions. *Journal of Geophysical Research: Atmospheres*, February 2022. ISSN 2169-897X, 2169-8996. doi:10.1029/2021JD035597. URL https://onlinelibrary.wiley.com/doi/10.1029/2021JD035597.

[27] Felix Kleinert, Lukas H. Leufen, Aurelia Lupascu, Tim Butler, and Martin G. Schultz. Representing chemical history in ozone time-series predictions – a model experiment study building on the MLAir (v1.5) deep learning framework. *Geoscientific Model Development*, 15(23):8913–8930, December 2022. ISSN 1991-9603. doi:10.5194/gmd-15-8913-2022. URL https://gmd.copernicus.org/articles/15/8913/2022/.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, Las Vegas, NV, USA, June 2016. IEEE. ISBN 978-1-4673-8851-1. doi:10.1109/CVPR.2016.90. URL http://ieeexplore.ieee.org/document/7780459/.

[29] CAMS. Regional Production, Updated documentation covering all Regional operational systems and the ENSEMBLE. *Regional air quality production systems*, Copernicus Atmosphere Monitoring Service, March 2020. URL https://atmosphere.copernicus.eu/sites/default/files/2020-09/CAMS50_2018SC2_D2.0.2-U2_Models_documentation_202003_v2.pdf.

[30] Martin G Schultz, Sabine Schröder, Olga Lyapina, Owen R Cooper, Ian Galbally, Irina Petropavlovskikh, Erika Von Schneidemesser, Hiroshi Tanimoto, Yasin Elshorbany, Manish Naja, et al. Tropospheric ozone assessment report: Database and metrics data of global surface ozone observations. *Elementa: Science of the Anthropocene*, 5, 2017. doi:10.1525/elementa.244.

[31] European Parliament and Council of the European Union. Directive 2008/50/ec of the european parliament and of the council of 21 may 2008 on ambient air quality and cleaner air for europe. *Official Journal of the European Union*, 2008. URL http://data.europa.eu/eli/dir/2008/50/oj.

[32] Lukas Hubert Leufen, Felix Kleinert, and Martin G. Schultz. Exploring decomposition of temporal patterns to facilitate learning of neural networks for ground-level daily maximum 8-hour average ozone prediction. *Environmental Data Science*, 1:e10, 2022. doi:10.1017/eds.2022.9.

[33] Hans Hersbach, Bill Bell, Paul Berrisford, Shoji Hirahara, András Horányi, Joaquín Muñoz-Sabater, Julien Nicolas, Carole Peubey, Raluca Radu, Dinand Schepers, Adrian Simmons, Cornel Soci, Saleh Abdalla, Xavier Abellan, Gianpaolo Balsamo, Peter Bechtold, Gionata Biavati, Jean Bidlot, Massimo Bonavita, Giovanna De Chiara, Per Dahlgren, Dick Dee, Michail Diamantakis, Rossana Dragani, Johannes Flemming, Richard Forbes, Manuel Fuentes, Alan Geer, Leo Haimberger, Sean Healy, Robin J. Hogan, Elías Hólm, Marta Janisková, Sarah Keeley, Patrick Laloyaux, Philippe Lopez, Cristina Lupu, Gabor Radnoti, Patricia de Rosnay, Iryna Rozum, Freja Vamborg, Sebastien Villaume, and Jean-Noël Thépaut. The era5 global reanalysis. *Quarterly Journal of the Royal Meteorological Society*, 146(730):1999–2049, 2020. doi:https://doi.org/10.1002/qj.3803. URL https://rmets.onlinelibrary.wiley.com/doi/abs/10.1002/qj.3803.

[34] TOAR Data Team. TOAR Data Infrastructure, 2023. Accessed 18 January 2023, https://toar-data.fz-juelich.de/.

[35] Copernicus Climate Change Service. ERA5: data documentation, 2022. Accessed 18 January 2023, https://confluence.ecmwf.int/display/CKB/ERA5

[36] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J. Leon Zhao. Time Series Classification Using Multi-Channels Deep Convolutional Neural Networks. In David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Alfred Kobsa, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Demetri Terzopoulos, Doug Tygar, Gerhard Weikum, Feifei Li, Guoliang Li, Seung-won Hwang, Bin Yao, and Zhenjie Zhang, editors, *Web-Age Information Management*, volume 8485, pages 298–310. Springer International Publishing, Cham, 2014. ISBN 978-3-319-08009-3 978-3-319-08010-9. doi:10.1007/978-3-319-08010-9_33. URL http://link.springer.com/10.1007/978-3-319-08010-9_33. Series Title: Lecture Notes in Computer Science.

[37] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[38] Alex Bauerle, Christian van Onzenoodt, and Timo Ropinski. Net2vis – a visual grammar for automatically generating publication-tailored cnn architecture visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 27(6):2980–2991, Jun 2021. ISSN 2160-9306. doi:10.1109/tvcg.2021.3057483. URL http://dx.doi.org/10.1109/TVCG.2021.3057483.

[39] ADS. CAMS European air quality forecasts, February 2020. URL https://ads.atmosphere.copernicus.eu/cdsapp#!/dataset/cams-europe-air-quality-forecasts?tab=overview.

[40] V. Marécal, V.-H. Peuch, C. Andersson, S. Andersson, J. Arteta, M. Beekmann, A. Benedictow, R. Bergström, B. Bessagnet, A. Cansado, F. Chéroux, A. Colette, A. Coman, R. L. Curier, H. A. C. Denier van der Gon, A. Drouin, H. Elbern, E. Emili, R. J. Engelen, H. J. Eskes, G. Foret, E. Friese, M. Gauss, C. Giannaros, J. Guth, M. Joly, E. Jaumouillé, B. Josse, N. Kadygrov, J. W. Kaiser, K. Krajsek, J. Kuenen, U. Kumar, N. Liora, E. Lopez, L. Malherbe, I. Martinez, D. Melas, F. Meleux, L. Menut, P. Moinat, T. Morales, J. Parmentier, A. Piacentini, M. Plu, A. Poupkou, S. Queguiner, L. Robertson, L. Rouïl, M. Schaap, A. Segers, M. Sofiev, L. Tarasson, M. Thomas, R. Timmermans, Á. Valdebenito, P. van Velthoven, R. van Versendaal, J. Vira, and A. Ung. A regional air quality forecasting system over Europe: the MACC-II daily ensemble production. *Geoscientific Model Development*, 8(9):2777–2813, 2015. ISSN 1991-9603. doi:10.5194/gmd-8-2777-2015. URL https://gmd.copernicus.org/articles/8/2777/2015/.

[41] Vincent-Henri Peuch, Richard Engelen, Michel Rixen, Dick Dee, Johannes Flemming, Martin Suttie, Melanie Ades, Anna Agustí-Panareda, Cristina Ananasso, Erik Andersson, David Armstrong, Jérôme Barré, Nicolas Bousserez, Juan Jose Dominguez, Sébastien Garrigues, Antje Inness, Luke Jones, Zak Kipling, Julie Letertre-Danczak, Mark Parrington, Miha Razinger, Roberto Ribas, Stijn Vermoote, Xiaobo Yang, Adrian Simmons, Juan Garcés de Marcilla, and Jean-Noël Thépaut. The Copernicus Atmosphere Monitoring Service: from research to operations. *Bulletin of the American Meteorological Society*, August 2022. ISSN 0003-0007, 1520-0477. doi:10.1175/BAMS-D-21-0314.1. URL https://journals.ametsoc.org/view/journals/bams/aop/BAMS-D-21-0314.1/BAMS-D-21-0314.1.xml.

[42] Allan H. Murphy and Robert L. Winkler. A General Framework for Forecast Verification. *Monthly Weather Review*, 115(7):1330–1338, July 1987. ISSN 0027-0644, 1520-0493. doi:10.1175/1520-0493(1987)115<1330:AGFFFV>2.0.CO;2. URL http://journals.ametsoc.org/doi/abs/10.1175/1520-0493%281987%29115%3C1330%3AAGFFFV%3E2.0.CO%3B2. Number: 7.

[43] Daniel S. Wilks. *Statistical Methods in the Atmospheric Sciences*. Academic Press, London, 2nd edition, 2006.

[44] Leo Breiman. Random Forests. *Machine Learning*, 45(1):5–32, 2001. ISSN 08856125. doi:10.1023/A:1010933404324. URL http://link.springer.com/10.1023/A:1010933404324.

[45] P. J. Young, V. Naik, A. M. Fiore, A. Gaudel, J. Guo, M. Y. Lin, J. L. Neu, D. D. Parrish, H. E. Rieder, J. L. Schnell, S. Tilmes, O. Wild, L. Zhang, J. Ziemke, J. Brandt, A. Delcloo, R. M. Doherty, C. Geels, M. I. Hegglin, L. Hu, U. Im, R. Kumar, A. Luhar, L. Murray, D. Plummer, J. Rodriguez, A. Saiz-Lopez, M. G. Schultz, M. T. Woodhouse, and G. Zeng. Tropospheric Ozone Assessment Report: Assessment of global-scale model performance for global and regional ozone distributions, variability, and trends. *Elementa: Science of the Anthropocene*, 6:10, January 2018. ISSN 2325-1026. doi:10.1525/elementa.265. URL https://online.ucpress.edu/elementa/article/doi/10.1525/elementa.265/112813/Tropospheric-Ozone-Assessment-Report-Assessment-of.

[46] Peter Bauer, Alan Thorpe, and Gilbert Brunet. The quiet revolution of numerical weather prediction. *Nature*, 525(7567):47–55, September 2015. ISSN 0028-0836, 1476-4687. doi:10.1038/nature14956. URL http://www.nature.com/articles/nature14956.

[47] Thomas Haiden, Martin Janousek, Frédéric Vitart, Zied Ben-Bouallegue, Laura Ferranti, Fernando Prates, and David Richardson. Evaluation of ECMWF forecasts, including the 2021 upgrade. *European Centre for Medium-Range Weather Forecasts*, 2022. doi:10.21957/XQNU5O3P. URL https://www.ecmwf.int/node/20469. Publisher: ECMWF.

[48] Dallas Foster, David John Gagne, and Daniel B. Whitt. Probabilistic Machine Learning Estimation of Ocean Mixed Layer Depth From Dense Satellite and Sparse In Situ Observations. *Journal of Advances in Modeling Earth Systems*, 13(12), December 2021. ISSN 1942-2466, 1942-2466. doi:10.1029/2021MS002474. URL https://onlinelibrary.wiley.com/doi/10.1029/2021MS002474.

[49] Elizabeth A. Barnes, Randal J. Barnes, and Nicolas Gordillo. Adding Uncertainty to Neural Network Regression Tasks in the Geosciences. *arXiv*, 2021. doi:10.48550/ARXIV.2109.07250. URL https://arxiv.org/abs/2109.07250. Publisher: arXiv Version Number: 1.

[50] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf`.

[51] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, October 2021. ISSN 01692070. doi:10.1016/j.ijforecast.2021.03.012. URL `https://linkinghub.elsevier.com/retrieve/pii/S0169207021000637`.

[52] James F Kaiser. Digital filters. In Franklin F Kuo and James F Kaiser, editors, *System analysis by digital computer*, chapter 7, pages 218–285. Wiley New York, NY, 1966.

[53] L. H. Leufen, F. Kleinert, F. Weichselbaum, V. Gramlich, and M. G. Schultz. MLAir – a tool to enable fast and flexible machine learning on air data time series, version 2.3.0, source code, 2022. URL `https://gitlab.jsc.fz-juelich.de/esde/machine-learning/mlair/`.

[54] L. H. Leufen, F. Kleinert, and M. G. Schultz. MLAir (v1.0) – a tool to enable fast and flexible machine learning on air data time series. *Geoscientific Model Development*, 14(3):1553–1574, 2021. doi:10.5194/gmd-14-1553-2021. URL `https://gmd.copernicus.org/articles/14/1553/2021/`.

[55] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL `https://www.tensorflow.org/`. Software available from tensorflow.org.

[56] L. Menut, B. Bessagnet, D. Khvorostyanov, M. Beekmann, N. Blond, A. Colette, I. Coll, G. Curci, G. Foret, A. Hodzic, S. Mailler, F. Meleux, J.-L. Monge, I. Pison, G. Siour, S. Turquety, M. Valari, R. Vautard, and M. G. Vivanco. CHIMERE 2013: a model for regional atmospheric composition modelling. *Geoscientific Model Development*, 6(4):981–1028, July 2013. ISSN 1991-9603. doi:10.5194/gmd-6-981-2013. URL `https://gmd.copernicus.org/articles/6/981/2013/`.

[57] Jesper Heile Christensen. The Danish eulerian hemispheric model — a three-dimensional air pollution model used for the arctic. *Atmospheric Environment*, 31(24):4169–4191, December 1997. ISSN 13522310. doi:10.1016/S1352-2310(97)00264-1. URL `https://linkinghub.elsevier.com/retrieve/pii/S1352231097002641`.

[58] D. Simpson, A. Benedictow, H. Berge, R. Bergström, L. D. Emberson, H. Fagerli, C. R. Flechard, G. D. Hayman, M. Gauss, J. E. Jonson, M. E. Jenkin, A. Nyíri, C. Richter, V. S. Semeena, S. Tsyro, J.-P. Tuovinen, Á. Valdebenito, and P. Wind. The EMEP MSC-W chemical transport model – technical description. *Atmospheric Chemistry and Physics*, 12(16):7825–7865, August 2012. ISSN 1680-7324. doi:10.5194/acp-12-7825-2012. URL `https://acp.copernicus.org/articles/12/7825/2012/`.

[59] H. Hass, H. J. Jakobs, and M. Memmesheimer. Analysis of a regional model (EURAD) near surface gas concentration predictions using observations from networks. *Meteorology and Atmospheric Physics*, 57(1-4):173–200, 1995. ISSN 0177-7971, 1436-5065. doi:10.1007/BF01044160. URL `http://link.springer.com/10.1007/BF01044160`.

[60] M. Memmesheimer, E. Friese, A. Ebel, H.J. Jakobs, H. Feldmann, C. Kessler, and G. Piekorz. Long-term simulations of particulate matter in Europe on different scales using sequential nesting of a regional model. *International Journal of Environment and Pollution*, 22(1/2):108, 2004. ISSN 0957-4352, 1741-5101. doi:10.1504/IJEP.2004.005530. URL `http://www.inderscience.com/link.php?id=5530`.

[61] J. W. Kaminski, L. Neary, J. Struzewska, J. C. McConnell, A. Lupu, J. Jarosz, K. Toyota, S. L. Gong, J. Côté, X. Liu, K. Chance, and A. Richter. GEM-AQ, an on-line global multiscale chemical weather modelling system: model description and evaluation of gas phase chemistry processes. *Atmospheric Chemistry and Physics*, 8(12):3255–3281, June 2008. ISSN 1680-7324. doi:10.5194/acp-8-3255-2008. URL `https://acp.copernicus.org/articles/8/3255/2008/`.

[62] Martijn Schaap, Renske M.A. Timmermans, Michiel Roemer, G.A.C. Boersen, Peter J.H. Builtjes, Ferd J. Sauter, Guus J.M. Velders, and Jeanette P. Beck. The LOTOS EUROS model: description, validation and latest developments. *International Journal of Environment and Pollution*, 32(2):270, 2008. ISSN 0957-4352, 1741-5101. doi:10.1504/IJEP.2008.017106. URL `http://www.inderscience.com/link.php?id=17106`.

[63] Lennart Robertson, Joakim Langner, and Magnuz Engardt. An Eulerian Limited-Area Atmospheric Transport Model. *Journal of Applied Meteorology*, 38(2):190–210, February 1999. ISSN 0894-8763, 1520-0450. doi:10.1175/1520-0450(1999)038<0190:AELAAT>2.0.CO;2. URL `http://journals.ametsoc.org/doi/10.1175/1520-0450(1999)038<0190:AELAAT>2.0.CO;2`.

[64] C. Andersson, R. Bergström, C. Bennet, L. Robertson, M. Thomas, H. Korhonen, K. E. J. Lehtinen, and H. Kokkola. MATCH-SALSA – Multi-scale Atmospheric Transport and CHemistry model coupled to the SALSA aerosol microphysics model – Part 1: Model description and evaluation. *Geoscientific Model Development*, 8(2):171–189, February 2015. ISSN 1991-9603. doi:10.5194/gmd-8-171-2015. URL `https://gmd.copernicus.org/articles/8/171/2015/`.

[65] B. Josse, P. Simon, and V. H. Peuch. Radon global simulations with the multiscale chemistry and transport model MOCAGE. *Tellus B: Chemical and Physical Meteorology*, 56(4):339–356, January 2004. ISSN 1600-0889. doi:10.3402/tellusb.v56i4.16448. URL `https://www.tandfonline.com/doi/full/10.3402/tellusb.v56i4.16448`.

[66] A. Dufour, M. Amodei, G. Ancellet, and V.-H. Peuch. Observed and modelled "chemical weather" during ESCOMPTE. *Atmospheric Research*, 74(1-4):161–189, March 2005. ISSN 01698095. doi:10.1016/j.atmosres.2004.04.013. URL `https://linkinghub.elsevier.com/retrieve/pii/S0169809504001498`.

[67] Mikhail Sofiev, Michael Galperin, and Eugene Genikhovich. A Construction and Evaluation of Eulerian Dynamic Core for the Air Quality and Emergency Modelling System SILAM. In Carlos Borrego and Ana Isabel Miranda, editors, *Air Pollution Modeling and Its Application XIX*, pages 699–701. Springer Netherlands, Dordrecht, 2008. ISBN 978-1-4020-8452-2 978-1-4020-8453-9. doi:10.1007/978-1-4020-8453-9_94. URL `http://link.springer.com/10.1007/978-1-4020-8453-9_94`. ISSN: 1871-4668 Series Title: NATO Science for Peace and Security Series.

BONNER METEOROLOGISCHE ABHANDLUNGEN

Herausgegeben vom Institut für Geowissenschaften der Universität Bonn, Abteilung Meteorologie, durch Prof. Dr. H. FLOHN (Hefte 1-25), Prof. Dr. M. HANTEL (Hefte 26-35), Prof. Dr. H.-D. SCHILLING (Hefte 36-39), Prof. Dr. H. KRAUS (Hefte 40-49), ab Heft 50 durch Prof. Dr. A. HENSE.

Heft 1-79: siehe https://www.ifgeo.uni-bonn.de/abteilungen/meteorologie/bibliothek/bonner-meteorologische-abhandlungen-bma

80-96: open access, verfügbar unter https://bonndoc.ulb.uni-bonn.de/xmlui/handle/20.500.11811/1627

Heft 80: ***Tanja Zerenner***: Atmospheric downscaling using multi-objective genetic programming, 2016, [erschienen] 2017, X, 191 S.

Heft 81: ***Sophie Stolzenberger***: On the probabilistic evaluation of decadal and paleoclimate model predictions, 2017, IV, 122 S.

Heft 82: ***Insa Thiele-Eich***: Flooding in Dhaka, Bangladesh, and the challenge of climate change, 2017, V, 158 S.

Heft 83: ***Liselotte Bach***: Towards a probabilistic regional reanalysis for Europe, 2017 [erschienen] 2018, VI, 114 S.

Heft 84: ***Yen-Sen Lu***: Propagation of land surface model uncertainties in terrestrial system states, 2017, [erschienen] 2018, X, 120 S.

Heft 85: ***Rüdiger Hewer***: Stochastic physical models for wind fields and precipitation extremes, 2018, 99 S.

Heft 86: ***Sebastian Knist***: Land-atmosphere interactions in multiscale regional climate change simulations over Europe, 2018, VIII, 147 S.

Heft 87: ***Jessica Keune***: Integrated terrestrial simulations at the continental scale: Impact of groundwater dynamics and human water use on groundwater-to-atmosphere feedbacks during the European heatwave in 2003, 2019, IX, 172 S.

Heft 88: ***Christoph Beekmans***: 3-D Cloud Morphology and Evolution Derived from Hemispheric Stereo Cameras, 2019, [erschienen] 2020, VIII, 118 S.

Heft 89: ***Nils Weitzel***: Climate field reconstructions from pollen and macrofossil syntheses using Bayesian hierarchical models, 2019, [erschienen] 2020, XII, 153 S.

Heft 90: ***Alexander Kelbch***: Investigations to quantify individual exposure to solar ultraviolet erythemal radiation including cloud meteorological impact, 2020, III, 107 S.

Heft 91: ***Mari L. Schmidt***: Improvement of hail detection and nowcasting by synergistic combination of information from polarimetric radar, model predictions, and in-situ observations, 2020, VI, 136 S.

Heft 92: ***Sebastian Brune***: Der Wavelet-basierte Organisationsindex als Maß der konvektiven Organisation über Deutschland und dem tropischen Atlantik, 2021, IV, 121 S.

Heft 93: ***Sebastian Buschow***: Spatial Verification with Wavelets, 2022, V, 195 S.

Heft 94: ***Michael Langguth***: Representation of deep convection at gray-zone resolutions - Implementing and testing the HYbrid MAss flux Convection Scheme (HYMACS) in the ICON model, 2022, VI, 173 S.

Heft 95: ***Timon Netzel***: Quantitative paleoclimate reconstructions in the European region based on multiple proxies, 2023, VI, 179 S.

Heft 96: ***Lukas Hubert Leufen***: Time Filter Assisted Deep Learning to Predict Air Pollution, 2023, IV, 156 S.

Institut für Geowissenschaften
Abteilung Meteorologie
Mathematisch Naturwissenschaftliche Fakultät
Universität Bonn