Geometric deep learning for medium-range weather prediction

Master Project in Microengineering

Submitted June 2020, in partial fulfillment of
the conditions for the award of the degree
MSc in Microengineering

Icíar Lloréns Jover

Supervised by **Gionata Ghiggi** and **Michaël Defferrard**

Under the direction of:
Prof. Pierre Vandergheynst
Signal Processing Laboratory LTS2, EPFL

External Expert:
Dr. Peter Düben
European Center for Medium-Range Weather Forecasting (ECMWF), Research Department

École Polytechnique Fédérale de Lausanne
School of Engineering (STI)

Acknowledgments

I would first like to thank my Masters thesis director, Prof. Pierre Vandergheynst, head of the LTS2 laboratory, for providing the resources and framework that allowed the completion of this work.

I would also like to warmly thank my supervisors, Michaël Defferrard and Gionata Ghiggi, whose insight steered me through this research. I thank them also for their enthusiasm, support and patience.

In addition, I would like to thank Dr. Peter Düben, the expert involved in the jury for this research project, for kindly taking the time of reviewing and evaluating this work.

I am gratefully indebted to Monika Feldmann and Étienne Vignon for their advice and insights which are in part responsible for the performance achieved in this thesis.

Finally, I must express my very profound gratitude to Jonathan Regev, Adrian Montano, Yassine Zouaghi, Émile Bourban, Jeanne Esteves, Ana de las Heras, and Cyril Pulver for reviewing my work and for their continuous encouragement throughout the completion of this Masters thesis.

Contents

Abstract

Medium-range numerical weather prediction (NWP) is crucial to human activities. Reliable weather forecasts allow better resource management and are essential for disaster preparation. Modern NWP models provide accurate medium-range forecasts, but they require some of the largest computing facilities currently available. Convolutional Neural Networks (CNNs) are data-driven architectures preserving the spatial relationship of data with a low number of learnable parameters, thus having the potential to relieve this computational cost. As CNNs were originally tailored for Euclidean domains, current data-driven NWP approaches are based on planar projections of the sphere, which induces important deformations. This work proposes a data-driven medium-range NWP approach using a CNN able to perform convolutions natively on the sphere and integrating the temporal dimension. The effect of static and dynamic atmospheric features, as well as of the temporal discretization, is tested on the predictive skill using a wide range of metrics. Experimental results show that our method is able to emulate complex patterns of common atmospheric fields, and that our predictions are able to accurately reproduce the seasonal cycle. Our model outperforms the global and weekly climatologies and the persistence baselines up to 5 days. In addition, our model produces state-of-the-art temperature predictions for short forecast lead times.

1 Introduction

Numerical Weather Prediction (NWP) consists in predicting the future state and evolution of the atmosphere based on the partial knowledge of current conditions such as temperature, humidity and pressure, observed at sparse locations around the globe. Since the behavior of the atmosphere is continuous in space and time, weather models must discretize the underlying known physics in order to numerically simulate the evolution of the atmosphere. As such, the quality of the NWP forecast strongly depends on the spatio-temporal discretization, the parametrization of the physical processes, and the availability and assimilation of the partially known current atmospheric conditions. The success of modern weather forecast models can equally be credited on improved data acquisition techniques and on more accurate atmospheric models, both of which are reliant on increased computation power (Coiffier, 2011). Today, the European Centre for Medium-Range Weather Forecasts (ECMWF) produces 15–30 day forecasts with a horizontal resolution of 30–60 km and 30-min time steps. Such forecasts are computed twice a day and around 40 billion grid-column calculations are performed in about 2.5 h. This demands some of the largest supercomputing facilities available (Bauer et al., 2015).

A particular case of NWP is medium-range prediction. A definition of medium range weather prediction has been actively sought since the 1960s. It is commonly described as ranging from two days up to two weeks and usually aims to describe the time range in which the mid-latitude weather retains some deterministic predictability (Lorenz, 1969; Palmer et al., 1989; Buizza and Leutbecher, 2015). Medium-range forecasting is more extended in time than nowcasting, which requires an extended area of influence. Indeed, forecasts beyond 5 to 7 days have an area of influence larger than one hemisphere. Medium-range forecasting is of great importance to policy-makers. Firstly, economic activities are becoming gradually more dependent on weather forecasts as efficient resource management relies on good predictions. Secondly, medium-range forecasting is useful for disaster preparation, since these forecasts can predict wildfires, winter storms, extreme rainfalls and floods, or dry spells (Thorpe, 2012).

With the increase in computational power and in the amounts of data in electronic form, another discipline has gained popularity: Deep Learning (DL) methods. DL methods have many advantages. First, they are flexible models that can automatically learn representations from the data. Second, they result in empirically good performance when data is plentiful. Finally, as discussed in Section 2.1, current weather forecasting models have a drawback: despite their prediction accuracy, their computational cost is exorbitant (Bauer et al., 2015). DL techniques could relieve this cost as they need to be trained only once, after which they can make unlimited weather predictions at a low computational cost (Reichstein et al., 2019).

Previous works focused on medium-range weather prediction using two approaches: direct or iterative prediction. In direct prediction, a network is provided with atmospheric features and is trained to predict those features at a specific forecast lead time. This requires training one network for each desired forecast lead time. Moreover, direct predictions do not ensure a realistic spatial and temporal continuity. Iterative predictions are created from networks trained to predict a shorter forecast lead time, e.g. a few hours. These short-term predictions are then fed back into the network to create the subsequent forecasts. This is attractive as only one CNN needs to be trained to predict any forecast lead time. Its main disadvantage is that the CNN output differs from the training data. This results in unpredictable outputs, hence increasingly unrealistic predictions as iterations are performed. Table 1 provides a summary of the previous works herein mentioned.

	Data	**Sphere parame-trization**	**Type of prediction**	**Region**	**Multi-step input / output**	**Multi-step training**	**Resolution**
Düben and Bauer (2018)	ERA5	Plane	Iterative	Global	No	No	Spatial: 6° Temporal: 1h
Scher (2018)	GEFS	Plane	Direct	165°W–60°E, 0°–80°N	No	No	Spatial:0.49° Temporal: -
Scher and Messori (2019)	PLASIM and PUMA	Plane	Iterative	Global	No	No	Spatial: 5.65°, 2.8° Temporal: 24h
Weyn et al. (2019)	CFS	Plane	Iterative	North hemisphere	Yes	Yes	Spatial: 2.5° Temporal: 6h
Rasp et al. (2020)	WeatherBench	Cylinder	Direct and iterative	Global	No	No	Spatial: 5.625° Temporal: 6h
Weyn et al. (2020)	ERA5	Cube	Iterative	Global	No	Yes	Spatial: 2° Temporal: 6h
Ours	WeatherBench	Sphere	Iterative	Global	Yes	Yes	Spatial: 5.625° Temporal: 6h

Table 1: Summary of previous works. Spatial resolution is only provided for iterative predictions. It represents the temporal spacing between forecasts.

Düben and Bauer (2018) predicted the geopotential height at 500 hPa (Z500) and the temperature at 2 meters above the surface (T2m). They proposed two different CNNs: a global network and a local network, the latter taking special care of the north and south poles. The model performances were compared to the ECMWF Operational Integrated Forecast System (IFS), to an IFS ran at a lower spatial resolution, and to the persistence forecast. Only the local network was able to beat the persistence, and none of the networks were able to outperform the low resolution IFS baseline (Düben and Bauer, 2018).

Another attempt at data-driven weather forecasting was presented by Scher (2018). The aim of this study was to show that a neural network could learn the dynamics of a simple climate model. The data used in this study was generated by a highly simplified atmospheric model assuming no seasonal or diurnal cycle, no surface elevation, and no ocean. The CNN used had a Unet architecture (see Section 4.3). The longest forecast lead time provided was 14 days. The proposed CNN was able to emulate the idealized atmospheric model, creating stable series of iterative forecasts.

Scher and Messori (2019) showed that a Unet architecture could also make realistic predictions on more complex idealized models, albeit with lower skill than in Scher (2018). Indeed, Scher and Messori (2019) were able to predict short term weather but unable to create stable long-term runs. This work suggested that more complex weather models as well as iterative predictions make the NN more prone to instabilities.

Weyn et al. (2019) predicted Z500 and the 700-300 hPa geopotential thickness ($\tau_{700-300}$) using a Unet architecture with a convolutional long short-term memory (LSTM) layer. They introduced the idea of predicting multiple time steps to constrain the model to produce long-term stable predictions. As such, the input and the output of the model are two time steps instead of one. The resulting forecasts outperform the climatology baseline for 5 days.

Rasp et al. (2020) published WeatherBench, a subset of the ERA5 archive designed to allow fast prototyping and benchmarking of data-driven weather forecasting models. This study also proposed a global evaluation metric enabling a direct comparison between different methods: a latitude-weighted Root Mean Squared Error (wRMSE). Finally, Rasp et al. (2020) presented several baselines: the Operational IFS ran at original and lower resolutions, the global and weekly climatologies, and the persistence. They also introduced two machine learning techniques serving as baseline scores: a linear regression model and a CNN model. Rasp et al. (2020) use periodic padding in the longitudinal dimension, making the CNN see the data as a cylinder in an effort to mimic the shape of the sphere. The results show that the CNN model performs best when trained to make direct predictions, whereas the iterative predictions diverge quickly. The direct predictions are able to beat both climatology baselines and the persistence baseline for up to 5 days, but are unable to perform better than the Operational IFS at any resolution (Rasp et al., 2020).

Weyn et al. (2020) proposed another solution using deep CNNs. The mapping from the original data onto a numerical grid was performed in a mass-conservative way. The grid itself is a major contribution of this work. Indeed, the sphere was modeled by a 6 faced cube so as to minimize distortions at the poles. Finally, the model was trained iteratively to mitigate instabilities during the iterative predictions. Within this framework, Weyn et al. (2020) predicted Z500, $\tau_{700-300}$, Z1000, and T2m. In addition, several Earth constants were included in the inputs: solar radiation, surface elevation, and a land-sea mask. Weyn et al. (2020) trained one network for each face of the cube. The underlying assumption is that every face of the cube is self-sufficient for medium-range weather prediction. This assumption is supported by the fact that forecasts below 5 days have an area of influence smaller than one hemisphere (Bengtsson, 1985). Weyn et al. (2020) claim that the results are indefinitely stable weather forecasts with good predictability skill. Although less accurate than the Operational IFS, their framework is able to beat climatology and persistence baselines as well as the low resolution IFS baselines.

None of the above described works used a spherical spatial setting. Some attempts at approximating the sphere were made in Rasp et al. (2020) and in Weyn et al. (2020), by using respectively a cylinder and a cube. However, those parametrizations still lead to deformations of the atmospheric fields. As stated in Section 2.2, CNNs are able to detect local patterns regardless of their position or orientation in the image. This requires the learning algorithm to be able to *move* within the image. Movement over a 2D image is described by translations, while movement over a 3D sphere is a 3D rotation. Constructing a CNN that works on the sphere requires replacing translations by 3D rotations. Cohen et al. (2018) and Esteves et al. (2019) achieved this goal by implementing exact convolutions on the sphere, but at a high computational cost. To tackle that, Jiang et al. (2019) presented a set of pre-engineered filters. Instead of learning all filter parameters, the model only needs to learn the constants weighting such filters. Despite the promising results achieved with very few parameters, the lack of flexibility might prevent finding optimal filters. A small number of solutions involved a compromise between exact, equivariant 3D convolutions and computational efficiency. An example of it is DeepSphere, introduced by Defferrard et al. (2020) and detailed in Section 4.4.1.

The object of the present work is to provide a data-driven framework for medium-range weather prediction by exploiting the spherical spatial setting. In particular, this study pairs the dataset WeatherBench with the convolutional kernels presented in DeepSphere to examine the effects of architecture, atmospheric fields and the inclusion of time sequences on the model's predictability. The performance of our model is evaluated using various metrics and is benchmarked against climatological baselines, previous works and the operational IFS model.

2 Background

2.1 Numerical weather prediction

Even though NWP's beginnings can be traced back to the early 1900s, the discipline started gaining popularity only in the second half of the 20th century. Advances in NWP followed the development of computers during World War II, as the discipline requires high computational power. Moreover, the accuracy of weather forecasts dramatically increased with the advent of satellite measurements of atmospheric conditions (Coiffier, 2011).

The first reflections on NWP reportedly came from Bjerknes, who published in 1904 that weather forecasting is a deterministic problem that can be solved. The necessary conditions for such task were said to be: 1) sufficient representation of the atmosphere's initial conditions; and 2) sufficient knowledge of the laws governing the atmosphere's behaviour. The shortcomings of this discipline were also noticed early: NWP requires solving nonlinear partial differential equations for which there is no analytical solution (Bjerknes, 1904). In 1922, Richardson was the first to find a numerical solution to those equations by assuming a constant velocity of atmospheric fluids over time. His formulations are nowadays called primitive equations. Richardson computed a 6 hour ahead forecast by hand, however achieving unrealistic results. He estimated that he needed 64000 people performing computations to forecast the entire globe (Richardson, 1922). Courant, Friedrichs, and Lewy proposed in 1928 a method to solve differential equations with finite differences while specifying which constraints to comply with when discretizing. This was the first attempt at discretization (Courant et al., 1928). In 1946, the first electronic computer, the Electronic Numerical Integrator and Computer (ENIAC) became operational. This marked a milestone in the NWP evolution, as computations could now occur as large-scale, automatized processes. Indeed, in 1950, Charney, Fjørtoft and von Neumann achieved the first numerical prediction computed on ENIAC, yielding encouraging results for the geopotential 500-hPa prediction (Charney et al., 1950). Their work is seen as the starting point of modern numerical prediction (Platzman, 1979).

Since then, the increase in computational power allowed to increase the geographical area of weather forecast computations. Increasing the area is necessary to extend the time range of the forecasts. Consequently, models for limited geographical areas were replaced by hemispheric models, then finally by global models which allowed handling of interactions between hemispheres. For short range forecasting, limited area models (LAM) are still currently in use and their precision ha been increased by introducing finer grids (Coiffier, 2011).

Another dimension explored to make the atmospheric models more accurate was the incorporation of atmospheric physics. Such physical aspects are namely momentum, heat, and water vapor. They constitute the model's physics. An increasing number of physical processes were progressively incorporated into weather models. Smagorinsky declared that the atmospheric water cycle and its associated energy exchanges were key to understanding the atmosphere (Smagorinsky, 1962). The water vapour exchange between air and soil implies computing turbulence at surface level (Businger et al., 1971; Deardorff, 1972; Louis, 1979). This relies on additional variables such as surface temperature and soil cover (Deardorff, 1977). The effect of solar radiation was also incorporated using features such as the hour of the day and the cloud cover (Rodgers and Walshaw, 1967; Katayama, 1972). Nevertheless, these physical processes are typically unresolved and their interaction with the atmospheric equations needs to be parametrized. Consequently, these equations are often simplified.

In addition to allowing the inclusion of external factors, computational power also allowed to change the nature of the atmospheric models used. Richardson's primitive equations are easy to implement. However, they make strong assumptions on the atmospheric fluids, as they model them to be static. Tanguay et al. (1990), Laprise (1992), and Bubnová et al. (1995) moved from hydrostatic to dynamic equations allowing to simulate atmospheric motion.

Nowadays, modern operational weather forecasts consist of a complex process chain. The first part of the process is the acquisition of meteorological data. Following through, an objective analysis is performed, called data assimilation. The goal of global data assimilation systems is to recreate an atmospheric state as close as possible to the current reality. It is computed by taking into account the observations as well as the model and physical constraints, filling with simulations the places in the atmosphere where observational data is lacking. The third part is the forecasting process using models and the determination of weather parameters at local scale (Coiffier, 2011). It is important to know that the previously mentioned data assimilation systems are used to provide atmospheric initial conditions and consist in the current best guess of the global atmospheric state. These gridded data sets are called reanalyses (Warner, 2010).

2.2 Data-driven methods

Data-driven methods are a set of methods that can automatically detect patterns in data, and then use these patterns to predict future data, classify data or perform other decision-making tasks (Murphy, 2012). Our task is to predict data (future weather) given some inputs (current and past weather). We are consequently interested in supervised learning, which is a learning discipline where the goal is to learn a mapping from inputs X to outputs Y, given a labeled set of input-output pairs.

Supervised statistical learning involves building a statistical model for predicting an output based on one or more inputs. We assume that the inputs X and the outputs Y must have a certain kind of relationship. This can be written as $Y = f(X) + \epsilon$ where ϵ is a random error term. The function f that connects the input variable to the output variable is in general unknown. The goal of statistical learning is hence to estimate f (James et al., 2013). In statistical supervised learning, the protocol is to divide the whole dataset into training and testing sets. The dataset composed of pairs (X, Y) used by the model to learn f is called the training set. The model's performance is then evaluated on the testing set, where the estimated $\hat{f}$ is applied to $X_{testing}$ to produce estimated $\hat{Y}$. For example: we would like to estimate how temperature affects humidity. We are given a set of points in which we observe that humidity increases with temperature. We could decide to model the relationship between temperature and humidity as linear. We then need to know which is the slope and offset of the affine function that best fits the data points. Those two parameters are *learnt* from the *statistics* of the available training data (e.g. its mean, variance), hence giving the domain its name. However, the learning of the slope and offset comes after we have decided that the relationship is linear. What happens if the relationship is not a line, but a parabola instead? The success of model-based approaches is indeed dependent on which model we have decided to use on the data. Moreover, the more complex the data and the relationship is, the more complex the model needs to be, increasing the amounts of engineering.

Deep learning (DL) methods offer an alternative to model-based learning. DL relies on a specific kind of functions called artificial neural networks (ANNs). ANNs can construct multiple levels of representation by composing simple but non-linear modules, each transforming an input into a representation at a higher, more abstract level (LeCun et al., 2015). According to the universal approximation theory, with the aggregation of enough transformations, any complex function can be learnt (Csáji, 2001). We take interest in a special kind of neural networks called Convolutional Neural Networks (CNNs). CNNs are notorious for recognizing patterns regardless of their location. This property is desirable in NWP problems as the same atmospheric pattern could be found at different locations on Earth. Section 4.3 further expands on this topic.

3 Data

3.1 WeatherBench

ERA5 is the fifth generation of ECMWF global climate reanalyses. Weather reanalyses provide a numerical description of the recent global climate by combining weather forecast models with meteorological observations. Specifically, ERA5 benefits from numerous innovations with respect to its predecessor

ERA-Interim. Such innovations are namely increased temporal and spatial model resolution, a measure of the uncertainty of the climate state, and an improved physical model. Indeed, ERA5 is based on a newer Integrated Forecast System, the IFS Cy41r2, which incorporates 10 years of research and improvement on its components (atmosphere, land, ocean waves, and observations). This makes ERA5 one of the best guesses of the atmospheric state at any point in time (Hersbach and Dee, 2016; Hersbach et al., 2020). The raw data are available hourly from 1979 to present on a 0.25° latitude-longitude grid (721×1440 grid points) at 37 vertical pressure levels (C3S, 2017). ERA5 contains 344 single-level and multi-level atmospheric fields describing the atmosphere and the surface of the Earth (Hennermann, 2020).

Since the ERA5 dataset sizes several petabytes, Rasp et al. (2020) regridded a subset of the ERA5 variables to three lower resolutions using a bilinear interpolation: 5.625° (32×64 grid points), 2.8125° (64×128 grid points) and 1.40525° (128×256 grid points). Vertical pressure levels were reduced to 10 levels. The temporal resolution was kept to 1 hour, but only the years 1979 to 2018 were considered. Finally, the number of atmospheric fields was reduced to 19. Such subset of ERA5 has been named WeatherBench. WeatherBench is the first attempt at creating a common dataset which enables the comparison of different forecasting methods. The introduction of WeatherBench provides the machine learning community a foundation for accelerated research on NWP applications (Rasp et al., 2020).

The three-dimensional atmospheric variables included in WeatherBench are temperature, relative and specific humidity, vertical and horizontal wind components, geopotential, vorticity and potential vorticity. The geopotential $\Phi[\mathrm{m}^2\mathrm{s}^{-2}]$ at a certain pressure level p is proportional to the height z of the pressure level p. In fact, it is defined as the mechanical work that must be done against the Earth's gravitational field to raise a mass of 1 kg from sea level to that point. The geopotential $\Phi(z)$ at height z is thus given by

$$\Phi(z) = \int_0^{z \text{ at } p} g \, dz', \tag{1}$$

where $g = g(z')$ is the acceleration due to gravity on Earth (Wallace and Hobbs, 2006). However, in the lower atmosphere, $g \approx g_0 = 9.81\mathrm{m\ s}^2$ where g_0 is the globally averaged acceleration due to gravity at the Earth's surface. Potential vorticity takes into account the height of a fluid column as well as its local spin (vorticity). If a column is shortened and flattened, then it must spin more slowly. On the other hand, if a column is stretched and thinned (preserving mass), it should spin more quickly due to conservation of angular momentum (Talley et al., 2011). Single level fields included in WeatherBench are the temperature at 2 meters above the surface, wind at 10 meters above the surface, total cloud cover, precipitations and top-of-atmosphere solar radiation. Finally, the dataset contains a series of static features characterizing the Earth surface such as the surface type (land/water), the soil type, the surface elevation or orography, the latitudes, and the longitudes.

The present work makes only use of the coarsest resolution data provided by WeatherBench, i.e. 5.625° (32×64 grid points). This allowed a reasonable computational cost and fast prototyping, as well as direct comparison to previous works. Indeed, one aim of this work is to compare the performances of our model to the baselines presented by Rasp et al. (2020).

3.2 Features

We divide the atmospheric fields in WeatherBench into two kinds of features. We call the first kind static features and define them as the input atmospheric fields that do not need to be predicted. These features are all of the constant features described in 3.1 and the top-of-atmosphere radiation. Although top-of-atmosphere radiation is not a constant field, we consider it static as its evolution is well understood and can be accurately modelled. Since those fields do not need to be predicted, they are not included in the network's output. These features are expected to provide further information on the underlying physical processes. At worst, their inclusion does not change the network's predictive skill. The second kind are the dynamic features. These features are continuously changing and their evolution cannot be

modeled using simple equations, unlike the top-of-atmosphere radiation.

Ideally, in order to have a complete model of the atmosphere, all of the features should be used for weather prediction. However, using all of the features implies a very high computational cost. Consequently, the features used for prediction are hand-picked in this work.

3.3 Model validation

We divide WeatherBench into three sets: a training set, on which the networks are trained; a validation set used to tune model hyperparameters; and a testing set, on which the evaluation of the model is performed. We selected the period 1979 to 2012 for the training set, 2013 to 2016 for the validation set and 2017 to 2018 for the evaluation. The entire dataset is divided so as to prevent overlapping between sets following the methodology adopted by Rasp et al. (2020). The geopotential at 500 hPa (Z500) and the temperature at 850 hPa (T850) have been chosen as primary verification fields. Two reasons motivate this choice. Firstly, Z500 and T850 are two common, relevant fields in weather forecasting. Z500 is a standard verification variable for medium-range NWP, and T850 is relevant for human activity as it describes the temperature near the surface of the Earth. Second, Z500 and T850 have been used as verification variables also by Rasp et al. (2020) and Weyn et al. (2020). To evaluate the performance of our models, we compare them to three common baselines: persistence, global climatology and weekly climatology. The persistence forecast uses initialization fields as forecasts, the global climatology is a single mean over all samples in the training set, and the weekly climatology is a mean computed for each week of the year. A useful forecast should outperform both the persistence and the weekly climatology (Rasp et al., 2020).

4 Method

4.1 Autoregressive models

Autoregressive models are a well-known statistical approach in the field of time series forecasting. The term autoregression indicates that it is a regression of the variable against itself. An autoregressive model forecasts a current feature x_t using a linear combination of past feature values. The order of an autoregression L is the number of past values used to regress the current value (Hyndman and Athanasopoulos, 2018). The relationship between x_t and previous time steps can be defined as

$$x_t = \sum_{i=0}^{L} w_i x_{t-i} = W \left[x_{t-1}, x_{t-2}, ..., x_{t-L} \right]^T , \tag{2}$$

where W is the matrix of learnable model parameters. We refer to this as an AR(L) model, an autoregressive model of order L.

Equation (2) implies previous time steps are all of the consecutive values preceding x_t on discrete time series. However, values could be spaced of any arbitrary amount Δ_t describing the forecast lead time. Equation (2) can then be rewritten as:

$$x_t = \sum_{i=0}^{L} w_i x_{t-i\Delta_t} = W \left[x_{t-\Delta_t}, x_{t-2\Delta_t}, ..., x_{t-L\Delta_t} \right]^T . \tag{3}$$

Weather forecasting consists in predicting future weather based on present and past weather. The problem is closely linked to autoregressive models.

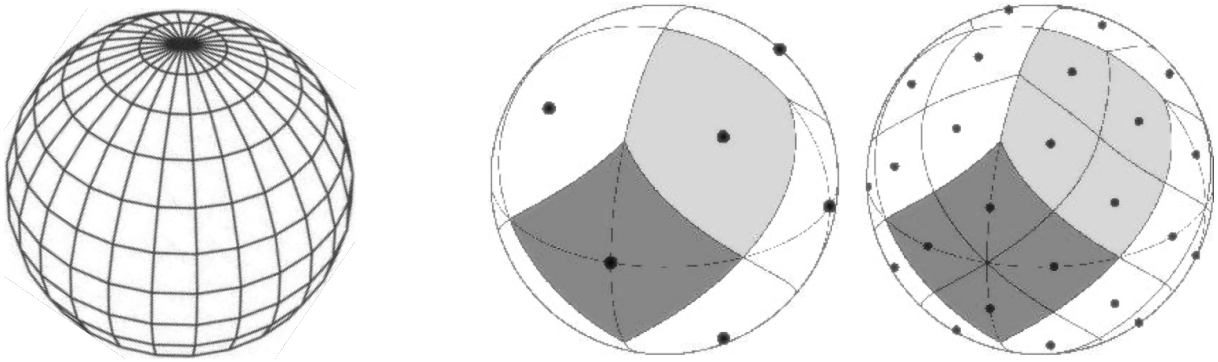
4.2 Spatial discretization

4.2.1 HEALPix sampling

An ideal spherical sampling respects three essential properties. The first one is hierarchical sampling: each pixel subdivision produces an equal number of child sub-pixels. This property is essential for pooling and unpooling operations, where the space is subdivided and regrouped. The second property is isolatitude. This indicates that the partition of the sphere is a collection of rings of pixels having the same latitude. This property is essential for computational speed of the spherical harmonic transform (Gorski et al., 1999). The last property is equal area pixels.

The sampling scheme used in WeatherBench, where samples are taken at a fixed resolution on the latitude and longitude grid, is called equiangular (Driscoll and Healy, 1994). Equiangular sampling has two of the three properties above listed: hierarchical sampling and isolatitude. However, equiangular sampling produces pixels that do not represent the same area and results in images where the poles are oversampled and the equator is undersampled. In addition to that, non-equal area sampling is region-dependant (see Figure 1a).

To address this problem, we use the HEALPix sampling. HEALPix — the Hierarchical Equal Area isoLatitude Pixelization — is a spherical sampling introduced by Gorski et al. (2005). It is a curvilinear partition of the sphere into equal area quadrilaterals of varying shape. The base subdivisions are 12 pixels, shown in the left sphere in Figure 1b. The resolution of the grid is expressed by the parameter N_{side} which defines the number of divisions along the side of the base-resolution pixel. The right sphere in Figure 1a shows a subdivision example where $N_{side} = 2$. The number of pixels resulting from a HEALPix sampling is $N = 12\ (N_{side})^2$. For HEALPix to remain a hierarchical sampling, N_{side} must be a power of 2. HEALPix is the only spherical sampling that respects all three properties: hierarchical, equi-area pixels and isolatitude (Gorski et al., 1999).



(a) Equiangular sampling

(b) HEALPix sampling

Figure 1: Orthographic projection of the equiangular and HEALPix samplings.

4.2.2 Spatial interpolation

As discussed in Section 3, this work uses the WeatherBench data at 5.625° resolution, which corresponds to $32 \times 64 = 2048$ pixels. When choosing an appropriate N_{side} for our HEALPix sampling, we decided to set $N_{side} = 16$, consequently fixing the number of pixels N to $N = 3072$. Indeed, choosing $N_{side} = 8$ would lead to $N = 12 \dots (8)^2 = 768$ pixels, which is an excessively low resolution. Moreover, as our regridding is done directly from WeatherBench instead of ERA5, having more pixels does not add any additional information.

The interpolation was done using the Python package SciPy through the xarray dataset interface. Two different kinds of interpolation were used. The range of the latitudes in WeatherBench is $[-87.1875, 87.1875]$ and the longitude range is $[0, 354.375]$. In HEALPix, for $N_{side} = 16$, the latitude range is $[-87.075, 87.075]$, which is included in the WeatherBench grid. However, the longitude range is $[0, 357.1874]$, which is outside the convex hull of the WeatherBench grid. We used linear interpolation for points inside the WeatherBench grid's convex hull and nearest neighbor interpolation for the points outside of it.

4.3 NN architecture

NNs learn to map input data to a target output. The most straightforward way to do so is by multiplying the input by a matrix containing one parameter describing the interaction between each input unit and each output unit. However, if we were to limit ourselves to stacking up blocks of linear transformations, called layers, the output of the network would be a linear function of its input. The goal of introducing NNs is to to replace W in the autoregressive model presented in Section 4.1 by a more expressive function. To tackle that, a non-linearity is introduced after each linear transformation. In DL, this non-linearity is called the activation function. Using an activation function on a linear transformation allows modelling nonlinear relationships between the input and the output (Goodfellow et al., 2016). Indeed, the universal approximation theorem states that a NN with a single hidden layer containing a finite number of neurons and a nonlinear activation can approximate any arbitrary real-valued function (Csáji, 2001).

In this work, we use a specialized kind of network called CNN. CNNs typically consist in 4 blocks: convolutions, activation, pooling and batch normalization.

Convolutions The matrix multiplication process described above is costly, as using a matrix to relate the input to the output means that every output unit interacts with every input unit. In addition to that, using a dense matrix to relate the input and the output makes the output dependent on the specific locations of the patterns in the input. CNNs provide a solution to those issues. CNNs make the parameter matrix, called kernel, much smaller than the input or the output. In order to do that, the kernel must interact with all pixels in the input sample, which can be done using an operation called convolution. Convolution can be seen as displacing the kernel in the signal domain to apply it to every part of the input. This operation effectively reduces the number of learnable parameters. In addition, convolution makes CNNs translation equivariant – if the input changes, the output changes in the same way. This property guarantees that the computations don't depend on an arbitrary coordinate system which gives CNNs more robustness (Goodfellow et al., 2016). Finally, translation equivariance removes the need for translational data augmentation which reduces the required size of the training dataset.

Activation The activation is an element-wise function. A wide range of activation functions, such as tanh or sigmoid, are used by the DL community. In modern neural networks, the most popular one is the Rectified Linear Unit, or ReLU (Jarrett et al., 2009; Nair and Hinton, 2010; Glorot et al., 2011). It is defined by:

$$g(z) = \max\{0, z\}. \tag{4}$$

ReLU's advantages are related to the gradient-based optimization of CNNs. First, ReLU is easy to differentiate as it is nearly linear. Moreover, since ReLU, unlike tanh or sigmoid, does not saturate, the span where its gradient is not zero is large: $g'(z) = 1, \ \forall z > 0$. Consequently, the inputs to the activation are less likely to fall within the range where the gradient tends to zero. As gradient-based optimization methods rely on the gradient to update the parameters, using ReLU makes the parameter update easier (Goodfellow et al., 2016). It is important to note that even though ReLU is the most popular activation function, it is not infallible. Rectified versions of ReLU, such as Leaky ReLU, have been found to outperform ReLU (Xu et al., 2015). However, the added complexity of implementing those functions and the lack of justification for their performance maintain ReLU as the go-to activation function.

Pooling A pooling function is a computational trick whose purpose is twofold. Firstly, it reduces the spatial dimensionality of a feature map by replacing a group of pixels by one pixel. The value of this replacement pixel is a summary statistic of the nearby pixels (Goodfellow et al., 2016). For instance, one of the most popular pooling operations is the max pooling, where the maximum value of a group of pixels is selected. Other alternatives are the average or a weighted average. By reducing the dimensionality of the input, each pixel becomes a representative of a group of pixels. This is useful for capturing more global patterns in the data (Defferrard, 2020). Secondly, pooling makes the output invariant to small deformations of the input. Indeed, whether an important feature is located a few pixels to the right or left is irrelevant as after the pooling only a summary of the neighborhood is kept. If pooling was performed over the input image, invariance over small transformations would ensue. However, when pooling over higher, more abstract representations of the data, the network can learn invariance to more complicated transformations (Goodfellow et al., 2016).

Batch normalization One of the challenges of training CNNs is that each layer's parameters need to be constantly re-updated at each training step to adjust to distribution shifts occurring in the previous layers. In order for the training to be stable, the learning rate needs to be consequently low. Ioffe and Szegedy (2015) address this problem by normalizing the inputs to each layer and making this normalization a part of the network's architecture. Batch Normalization (BN) allows to have higher learning rates, to be less careful with the network's initialization and tends to reduce overfitting (Ioffe and Szegedy, 2015).

Unet For our work, we chose to implement a variant of the notorious Unet architecture introduced by Ronneberger et al. (2015), an architecture initially implemented for semantic segmentation. The architecture consists in two blocks. The first one is the encoder. This block uses convolutions, BN and pooling operations in order to progressively reduce the spatial dimensionality of the feature maps while simultaneously increasing the number of feature maps. The encoder is used to capture context. Indeed, when the spatial dimensionality is reduced, pixels are forced to represent a larger area of the original image. Moreover, the increasing number of features allows for the more complex detection of patterns in data.

The second block is a decoder usually symmetric to the encoder. The decoder takes the most abstract feature maps and progressively reduces the number of features while increasing the spatial dimensionality. The output of the network has the same spatial dimensionality as the input. The drawback of the encoder is that, because of the pooling, a lot of spatial detail is lost. To tackle that, Ronneberger et al. (2015) added skip connections copying the tensor obtained at stage of encoder and concatenating it to its corresponding stage in the decoder. The decoder learns a combination of the encoded features and the inputs at different resolutions.

The Unet architecture is well suited for our task. Firstly, the output and the input have the same spatial shape, which is desirable as both our inputs and our outputs are atmospheric fields. Secondly, we are trying to capture the dynamics and movements of atmospheric states. To do so, our model needs to be able to understand the spatial interactions of the fields while keeping the high-resolution patterns in such fields. Unet's ability to capture context while maintaining spatial details is consequently attractive for prediction.

4.4 Tailored convolutions

4.4.1 Spherical convolutions

Section 4.3 describes convolutions designed to work with images. Indeed, a kernel can only be translated when the input samples are planar. Planar CNNs have been used to predict weather in past research by projecting atmospheric features onto 2D images. In this work, we aim at eliminating the deformations ensuing from projecting a sphere onto a plane by representing spherical data as a sphere while using a CNN architecture. Thus, this section introduces convolutional kernels that aim for 3D rotation equivariance

instead of 2D translational equivariance.

DeepSphere, a method introduced by Defferrard et al. (2020), addresses the aforementioned issue. DeepSphere is a graph-based method of representing discrete spherical data and a discrete spherical convolution for CNNs. Graphs are generic data representation forms which are useful to describe data of arbitrary shape. They are composed of two elements: the nodes (also called vertices) and the edges, which link the nodes together. The similarity between two nodes is usually represented by the weight of an edge: the larger the weight, the more similar the nodes (Shuman et al., 2013). In DeepSphere, the nodes represent the pixels of the input images which are connected according to the distance between them – greater distance implying less similarity.

Once the spherical discretization implemented, performing convolutions on the spherical domain is not trivial. Indeed, a filter cannot be moved in the discretized spherical domain without a uniform sampling. Moreover, the notion of shift does not exist for graphs (Defferrard et al., 2019). The solution to this issue is to apply the filters in the spectral domain. Given that a convolution is a multiplication in the spectral domain, no shifts need to be implemented. The kernels are hence replaced by low-order polynomials in the spectral domain. The exact implementation of spherical filtering costs $O(n^2)$ operations in general – the number of operations scales quadratically with the problem size – which can be prohibitive for weather and climate applications with over 10^9 pixels. DeepSphere strikes a controllable trade-off between 3D rotation equivariance and computational efficiency. See Defferrard et al. (2020) for more details on DeepSphere.

It is important to note that as spherical convolutions act on a circle with a parametrical diameter, the order of the polynomials K represents the diameter of the filter. Consequently, all of the kernel widths (convolutional or pooling) used with DeepSphere are one-dimensional and isotropic.

4.4.2 Temporal convolutions

Most of the recent data-driven NWP methods only consider an atmospheric state to be a snapshot of the atmosphere at a certain point in time, i.e. an autoregressive model of order $L = 1$. A single time step is however not enough to convey essential atmospheric information such as the speed or direction of fluid motion. Using several time-steps as input to the CNN is a solution to this problem. The order of the autoregression L is equivalent to the length of the input temporal sequence. In order to treat input data that has multiple time-steps, we perform convolutions in the temporal dimension. This is done by stacking L atmospheric snapshots, each snapshot consisting of several input features F_{in} along the feature dimension. The output of the network is only the subsequent snapshot. All of the snapshots are separated by a fixed Δ_t. Figure 2 shows how the temporal dimension is modeled in the network.

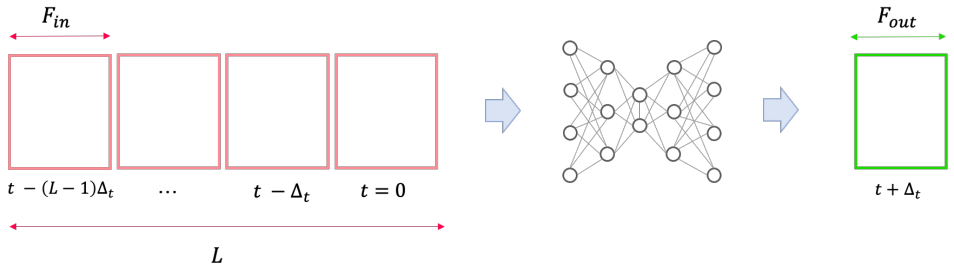


Figure 2: Temporal convolutions. L describes the number of input time steps in the input, which are separated by Δ_t. Each input time steps contains F_{in} features. The output of the network is a single time step containing F_{out} features. F_{out} is in general not equal to F_{in}, especially if the input contains static features that do not need to be predicted.

4.5 Our architecture

Our implementation combines a Unet architecture described in Section 4.3 with the spherical and temporal convolutions described in Section 4.4. Figure 3 shows our NN architecture.

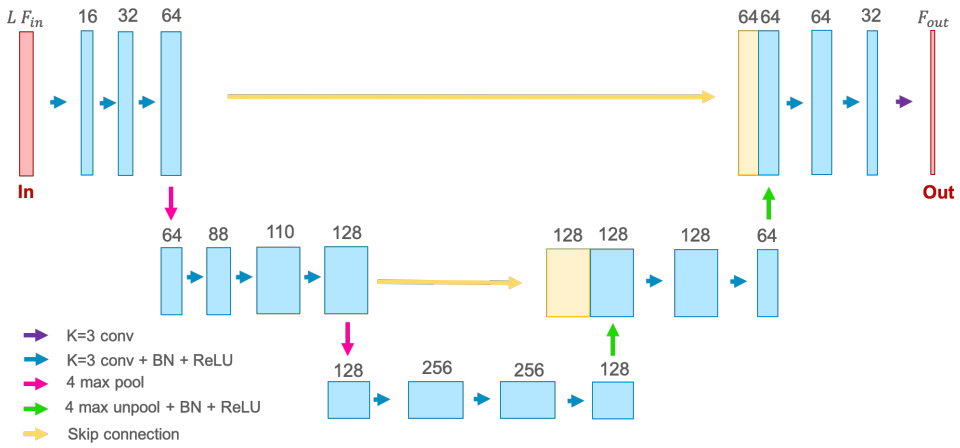


Figure 3: Network architecture. The Unet architecture consists of several blocks: convolutions, Batch Normalisation (BN), activation, pooling / unpooling modules and skip connections. This architecture retains all of the characteristics of the spatial-only architecture, as the temporal sequence is included in the features.The kernel diameter is K=3, the pooling is $2^2 = 4$ max pooling and the activation is ReLU. The temporal sequence is included in the dimension of the input features.

4.6 Training and evaluation

4.6.1 Loss function

All models were trained using a Mean Squared Error (MSE) loss function between each prediction and its corresponding observation. The terms p_{n} and o_{n} refer to the predicted and observed samples respectively. The MSE loss function is defined as

$$Loss = \mathrm{MSE}(p_{\mathrm{n}}, o_{\mathrm{n}}) = \frac{1}{N}\sum_{\mathrm{n}=0}^{N}(p_{\mathrm{n}} - o_{\mathrm{n}})^2, \tag{5}$$

where N is the number of samples in the training set.

4.6.2 Iterative predictions

The present work focuses solely on iterative predictions. This allows training a single network for all forecast lead times. Moreover, a network trained to produce iterative forecasts might contain more information on the underlying physics of the atmosphere than several networks trained for multiple forecast lead times. The predictions are performed iteratively given a certain starting point. Due to that, each sample in the evaluation set produces a four-dimensional tensor of forecasts, in which the first two dimensions are the space (latitudes and longitudes), the third dimension is determined by the number of features, and the fourth dimension is the number of iterations, i.e. the forecast lead times. Consequently, the entire evaluation set produces a five-dimensional tensor where the fifth dimension is the number of samples in the set. Figure 4 illustrates the shape of the predictions.

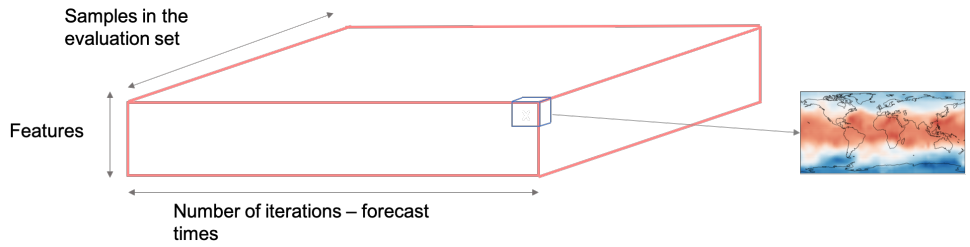


Figure 4: Schematic of the five-dimensional prediction tensor. The depicted three dimensions are: the number of features, the number of iterations – also symbolizing the forecast lead time, and the number of samples in the evaluation set. Each one of the entries of this schematic 3D matrix is a 2D image, making the tensor five-dimensional.

4.6.3 Iterative training

One of the problems of iterative predictions is that, as the number of iterations increases, the predictions grow more and more unrealistic and the predictive skill of the CNN decreases. This effect can be mitigated by training the CNN to minimize error on multiple iterated predictive steps using a multi time step loss function following Weyn et al. (2020). We call this iterative training. For our work, we have decided to limit the iterative training to two time-steps.

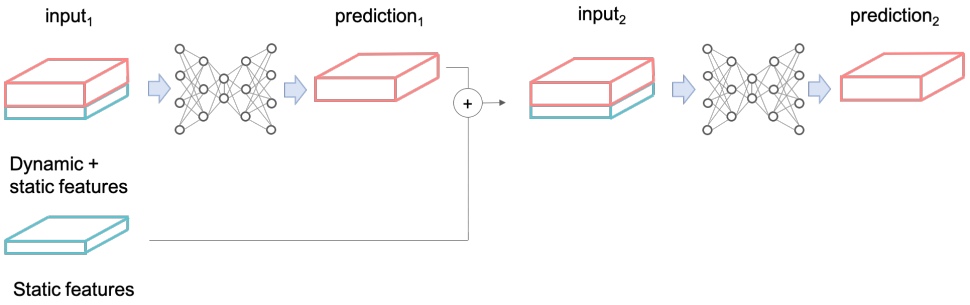


Figure 5: Iterative training. Here we have depicted the case where only two iterations are performed. Any number of iterations could be chosen. In that case, prediction$_2$ would be fed back into the CNN to produce prediction$_3$, and so on.

Iterative training goes as follows: the first input to the CNN is a combination of static and dynamic features. As static features do not need to be predicted, the CNN output is only the dynamic features of the next time step. In order to feed this output back to the CNN, the static features need to be re-integrated. This process is described in a schematic way by Figure 5.

Following the notation in Figure 5, the loss function in Equation (5) is rewritten as follows:

$$Loss = \alpha_1 \ \mathrm{MSE}(p_1,\ o_1) + \alpha_2 \ \mathrm{MSE}(p_2,\ o_2), \tag{6}$$

where and α_1, α_2 are two weighting coefficients. α_1 and α_2 were both set to 1 for simplicity, following Weyn et al. (2020).

4.6.4 Benchmarking

In order to assess the predictive skill of our CNNs and to compare them with the state-of-the-art, we evaluate them using the global metrics proposed by Rasp et al. (2020) alongside the dataset WeatherBench.

Such metrics are Root Mean Squared Error (wRMSE), Mean Absolute Error (MAE), and Anomaly Correlation Coefficient (ACC). They are defined as:

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{n=0}^{N}(p_\text{n} - o_\text{n})^2}, \tag{7}$$

$$\text{MAE} = \frac{1}{N}\sum_{n=0}^{N}|p_\text{n} - o_\text{n}|, \tag{8}$$

$$\text{ACC} = \frac{\sum_{n=0}^{N} p'_\text{n} o'_\text{n}}{\sum_{n=0}^{N} {p'_\text{n}}^2 \sum_{n=0}^{N} {o'_\text{n}}^2}, \tag{9}$$

where the prime $'$ indicates the subtraction of the observation's global climatology (Rasp et al., 2020).

These metrics were originally implemented to be applied to equiangular sampling. Consequently, Rasp et al. (2020) adapted them by weighting the pixels by the proportion of area it represents. Because of the spherical projection onto a plane, pixels at the poles represent a much smaller area than pixels at the equator. Consequently, polar pixels are given a smaller weight than the ones at the equator. In our case, the metrics do not need to be weighted, as all of the pixels in our model have the same area. Using these metrics, we compare the performance of our model to the direct and iterative predictions obtained with the CNN proposed by Rasp et al. (2020), as well as to several baselines also proposed in this work: persistence, global climatology, weekly climatology, and the operational IFS (Integrated Forecast System) model of the ECMWF. The last baseline is considered to be the gold standard of medium-range NWP.

4.6.5 Other evaluation metrics

The global RMSE metric does not allow to investigate in detail the strengths and weaknesses of our models. As a consequence, we introduce four more metrics to assess the ability of our CNNs to reproduce different aspects of the variable time series. They are defined using the work in Ghiggi et al. (2019). The skill metrics were computed on the test set, solely across the dimension describing the number of samples in the testing set. These metrics are consequently four-dimensional: the first dimension is the number of features, the second dimension is the forecast lead time and the last two dimensions are the space (see Figure 4). By not compressing the space into one statistic, we are able to assess the spatial distribution of the skills.

First, we introduce relRMSE, which represents the RMSE normalized by the observations. This skill allows us to localize in space and for each forecast lead time the relative error between the prediction and the observations. relRMSE is defined as follows:

$$\text{relRMSE} = \sqrt{\frac{\sum_{n=0}^{n_{samples}}(p_\text{n} - o_\text{n})^2}{\sum_{n=0}^{n_{samples}} o_\text{n}^2}}. \tag{10}$$

Secondly, in order to evaluate the overall tendency of the error magnitude, we introduce the relative bias (relBIAS). relBIAS has an optimal value of 0. A positive, respectively negative, value indicates a general overestimation, respectively underestimation, of the predictions with respect to the true data. It is defined as:

$$\text{relBIAS} = \frac{\text{mean}\,(p_\text{n} - o_\text{n})}{\text{mean}\,(o_\text{n})}. \tag{11}$$

relRMSE and relBIAS give us information about the magnitude and direction of the error. However, it is important to also assess if the predictions are correctly reproducing the variability of the atmospheric fields. The ratio of standard deviations (rSD) has an optimal value of 1. Values lower than 1 indicate underestimation, while values higher than 1 indicate overestimation of the observed variability. It is defined as:

$$\mathrm{rSD} = \frac{\mathrm{sd}\,(p_\mathrm{n})}{\mathrm{sd}\,(o_\mathrm{n})}. \tag{12}$$

The last metric used to assess our models is the squared correlation coefficient, R^2. It ranges between 0 and 1. It measures the degree of the linear association be- tween the predicted time series and the observed one. It is insensitive to the bias. The optimal value is 1.

Hereinafter, global values of the above mentioned skills refer to the their average value over all the spherical grid cells.

5 Experiments

In this section we describe the experiments that were performed on the data using the techniques detailed in Section 4, as well as their outcomes. Unless specified, all of the networks used an input sequence of $L = 1$, i.e. a single snapshot, and $\Delta_t = 6$.

5.1 Static features

In order to quantify how each one of the static features improve the network's predictive skill, we train 7 networks, each having different input features and all of them predicting the 500-hPa geopotential (Z500) and the 850-hPa temperature (T850). The inputs to these networks are:

- **No static features**: Z500 and T850
- **All static features**: Z500, T850, latitude, orography, land-sea mask, soil type, and top-of-atmosphere radiation
- **No latitudes**: Z500, T850, orography, land-sea mask, soil type, and top-of-atmosphere radiation
- **No orography**: Z500, T850, latitude, land-sea mask, soil type, and top-of-atmosphere radiation
- **No land-sea mask**: Z500, T850, latitude, orography, soil type, and top-of-atmosphere radiation
- **No soil type**: Z500, T850, latitude, orography, land-sea mask, and top-of-atmosphere radiation
- **No radiation**: Z500, T850, latitude, orography, land-sea mask, and soil type

These experiments are designed to show how each of the static features affects individually the predictability of the network. The interaction between the static features cannot be shown with this experimental design. Due to lack of space, since many models need to be compared, only global RMSE is used to compare them. Global relBIAS, rSD, and R2 followed the same trends as global RMSE.

The worst performing model is the one having no static features as input. However, despite our original hypothesis, the model having the lowest RMSE in the long-term is not the one having all of the static features. Indeed, even though the model with all static features is among the lowest RMSE curves, the Z500 prediction seems to benefit from removing the land-sea mask in the long term (Figure 6).

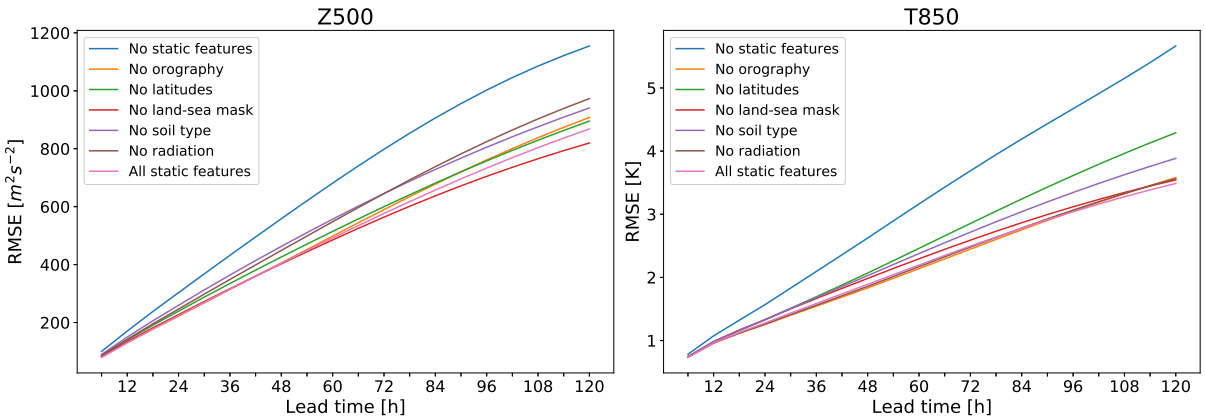


Figure 6: RMSE between the observations and the predictions of each one of the models as a function of the forecast lead time for Z500 and T850. First predictions benefit from the inclusion of more static features, but the land-sea mask makes the model diverge faster for Z500.

The best first prediction is given by the model with all static features, and the worst is given by the one with no static features (Figure 7). In early forecast lead times of the forecast, using more static features improves the predictive skill. In addition, removing one specific feature does not have the same effect for the prediction of Z500 or T850. Indeed, the feature improving the RMSE for Z500 the most is the soil type, whereas T850 is most benefited by the land-sea mask. For both Z500 and T850, including the orography does not improve much the RMSE at the first prediction.

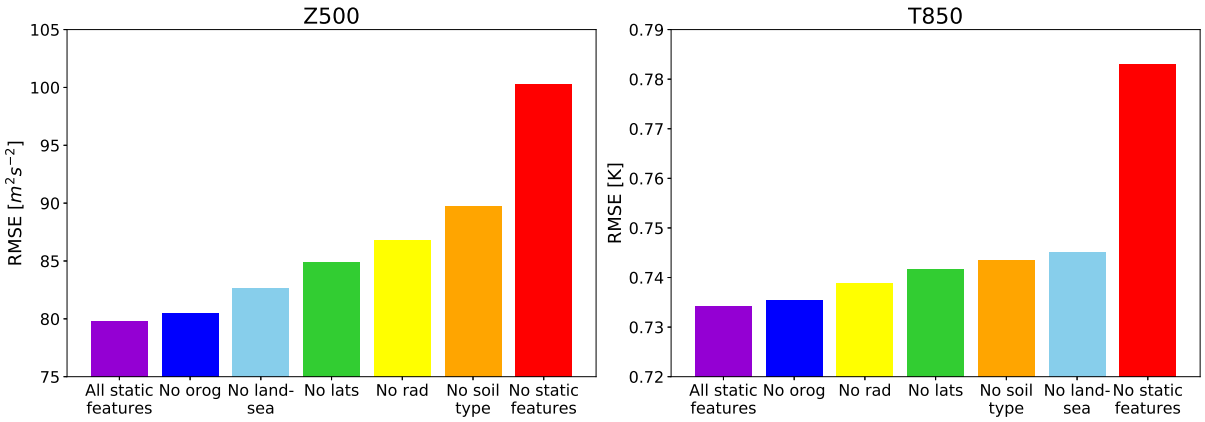


Figure 7: RMSE of first prediction (forecast lead time = 6 h), sorted in ascending order.

The tendencies observed for the first prediction change in the long-term. At 120 h forecast lead time, for T850, the lowest RMSE is still from the prediction issued by the model with all static features. It is not the case for Z500. Removing the land-sea mask lowers the RMSE of Z500 in the long term. Moreover, after 20 iterations, the features that lower the RMSE are not the same ones than for the first iteration. Indeed, for T850, including latitudes has the most effect, while for Z500, radiation is the most important feature for the long term (Figure 8).

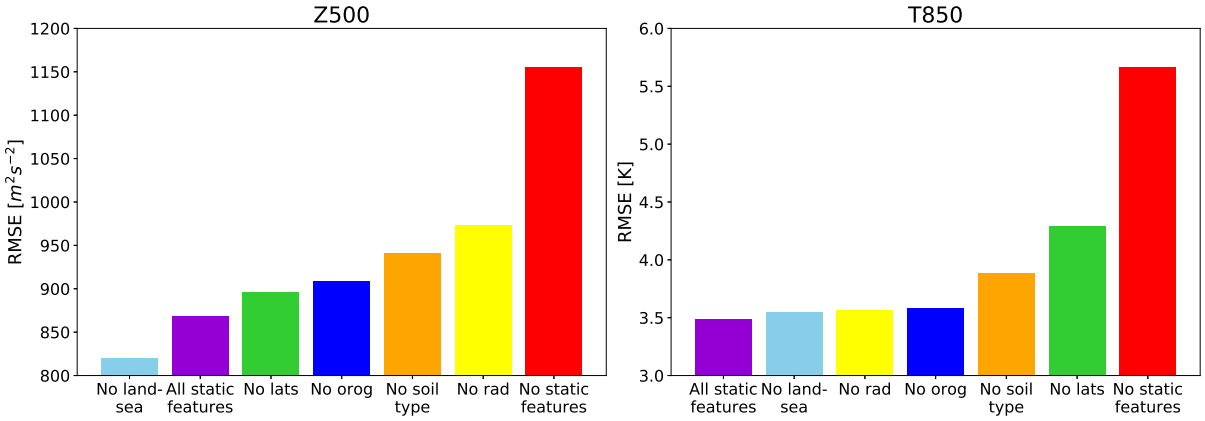


Figure 8: RMSE of last prediction (forecast lead time = 120 h), sorted in ascending order

All networks are trained to predict two consecutive time steps. The first predictions follow our expectations – using more static features lowers the error between predictions and observations. However, this statement does not apply to larger forecast lead times. This might be explained by the fact that, as iterations increase, errors in the predicted Z500 and T850 tend to accumulate. The static features are not predicted, but instead given to the network at each iteration. Consequently, the relationship that the network has learnt between the dynamic and static features might not be present, as Z500 and T850 diverge but the static features remain correct.

5.2 Dynamic features

The two atmospheric features that we predict in this work are Z500 and T850. Useful weather forecasts consist generally of other additional features, such as cloud cover or precipitation. In addition to that, some dynamic features might be useful to help with the prediction of either Z500 and T850, as they provide more information on the state and dynamics of the atmosphere. It is important to keep in mind that, because we are performing iterative predictions, those additional dynamic variables need to be predicted alongside Z500 and T850. The loss function of the CNN is conditioned on more features, and thus the predictability of the CNN is likely to decrease as dynamic features are added to the model.

5.2.1 Addition of Z1000

Following Weyn et al. (2020), we test the impact of Z1000 on the predictive skill of Z500 and T850. Since the vertical data sampling in WeatherBench is done by pressure levels instead of altitude levels, Z1000 is a proxy feature for surface pressure. As Z500 is affected mostly by the fluid density below 500 hPa (Feldmann, 2020, personal communication) including Z1000 might give useful information to the model regarding the evolution of Z500.

We train a single network having all of the static features as input, as well as Z500, T850 and Z1000. The output of this network are only the dynamic features Z500, T850 and Z1000. The predictions of this network are compared to the ones produced by a network having only Z500 and T850 as dynamic features. We used global RMSE and global R2 to evaluate the inclusion of Z1000 as a feature.

From the results obtained in Section 5.1, we expected the inclusion of Z1000 to at most improve first and second forecast predictions. The first RMSE is however increased by Z1000, although not significantly: the model with no additional dynamic features has a lower first RMSE than the model with Z1000 (see Table 2). Surprisingly, the main improvement in predictive skill resulting by adding Z1000 is in the iteration of Z500 predictions. The Z500 RMSE increase is slightly mitigated with the inclusion of Z1000.

On the contrary, this additional dynamic feature results in T850 predictions diverging more rapidly (Figure 9).

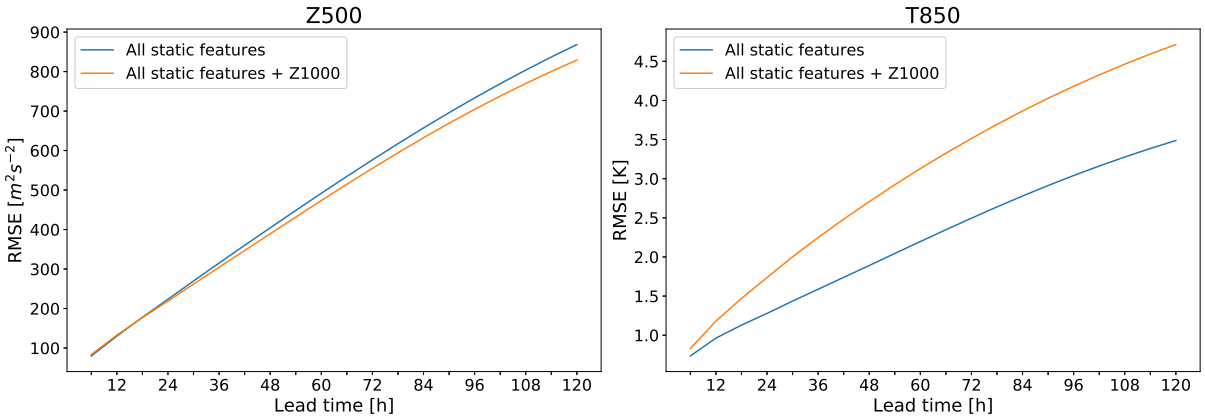


Figure 9: Global RMSE between predictions and observations for two models: all static features and all static features + Z1000. The RMSE is computed for each forecast lead time. Including Z1000 in the predictions lowers the Z500 RMSE slope and results in better long-term predictions. The effect is reversed for T850.

The T850 trend is reproduced when the correlation between predictions and observations is observed and measured by R2. Indeed, first predictions are similar between the two models. As forecast lead time increases, the predictions issued from the model including Z1000 become increasingly less correlated with the observations (Figure 10).

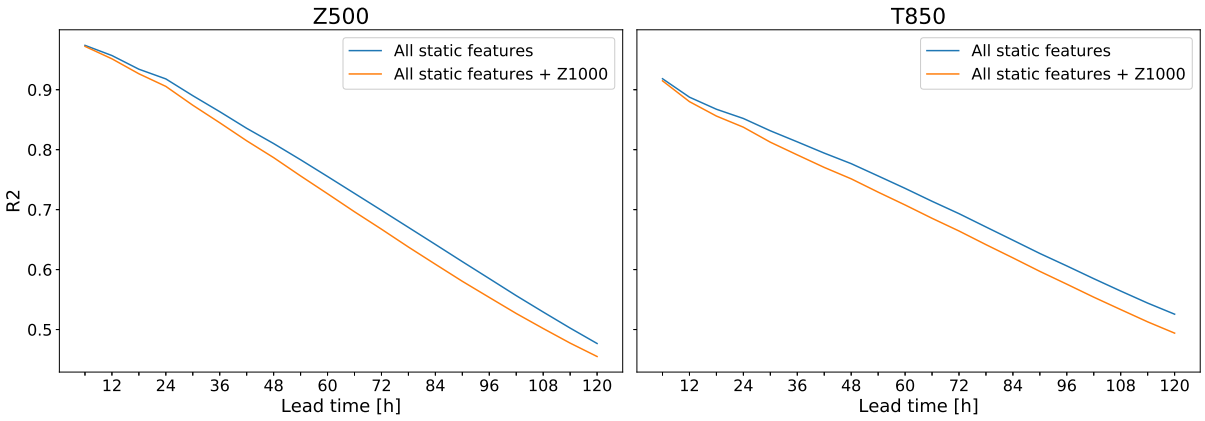


Figure 10: Global R2 measure between predictions and observations for two models: all static features and all static features + Z1000. R2 is computed for each forecast lead time. The inclusion of Z1000 results in the R2 decreasing faster with iterations.

The results of this experiment suggest that including geopotential features at pressure levels underneath 500 hPa might improve the Z500 RMSE in the long-term. Since Z1000 does not improve the T850 prediction, we deduct that Z1000 may not be as correlated with T850 as it is with Z500, hence resulting in faster divergence. The findings of this experiment motivate the inclusion of the vertical dimension in the predictions, as it shows that features at a certain pressure level are influenced by the other pressure levels.

5.2.2 Cloud cover and precipitation

This section of experiments aims at testing if cloud cover and precipitations can accurately be inferred using a data-driven NWP model. We train 3 networks with $L = 1$ and $\Delta_t = 6$, each having different input and output dynamic features but all having all of the static features as input:

- **Cloud cover**: the inputs to this network are all of the static features, Z500, T850 and the cloud cover. Its outputs are Z500, T850 and the cloud cover.
- **Precipitation**: the inputs to this network are all of the static features, Z500, T850 and the precipitations. Its outputs are Z500, T850 and the precipitations.
- **Could cover and precipitation**: the inputs to this network are all of the static features, Z500, T850, the cloud cover and the precipitations. Its outputs are Z500, T850, the cloud cover and the precipitations.

The aim of these experiments is twofold. First, we would like to see if their inclusion improves the Z500 and T850 predictions. This is assessed by comparing the resulting predictions with the predictions issued by the **All static features** model: its inputs being of the static features, Z500, T850, and its outputs Z500, and T850. For this task, only the global RMSE is used, as all of the other metrics follow the same tendency. Second, we would like to assess the predictive skill of our network on the cloud cover and the precipitation; first separately and then using their interaction. As we do not have any available benchmark for these predictors, the performance of the networks on such features is assessed using the spatial distribution of several skills.

Cloud cover and precipitation do not provide additional information to improve Z500 predictions. In fact, their inclusion, especially the one of cloud cover, makes subsequent iterations diverge faster (Figure 11). This can be explained by the fact that cloud cover and precipitation are the results of complex physical processes (i.e. convection,) occurring at spatio-temporal resolutions much smaller than the ones used in this study (5.625° and 6 hours). This essentially introduced noise in the training process, reducing also the ability of the network to predict Z500 accurately.

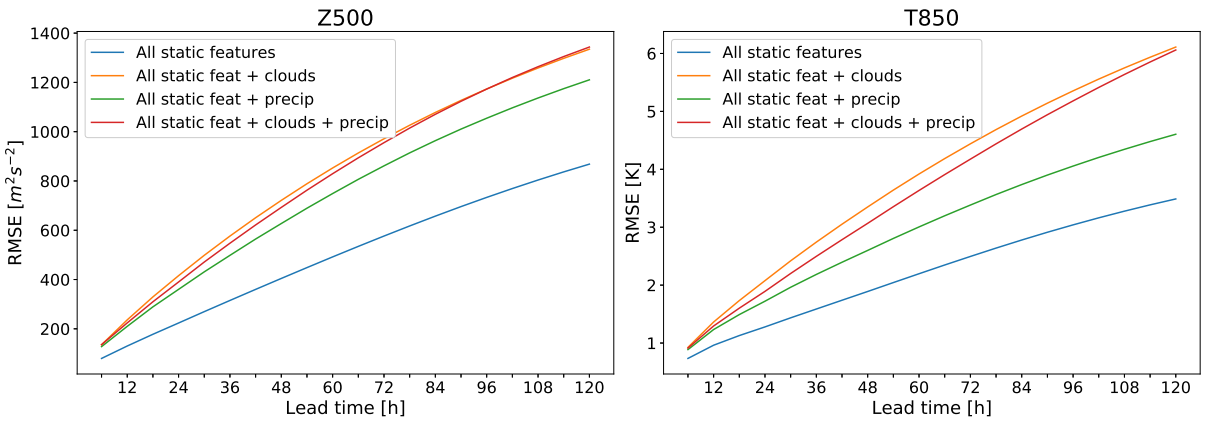


Figure 11: Global RMSE of Z500 and T850 between predictions and observations for four models. The RMSE is computed for each forecast lead time.

Even for a very short forecast lead time of 6 hours, our model cannot successfully predict the cloud cover or the total precipitation. Indeed, the relative RMSE is above 100% across the entire sphere for precipitations, and above 10% for the cloud cover. The direction of the cloud cover error is not explainable by latitude nor by land/sea partition. Instead, the model either greatly overestimates or underestimates the feature bias. On the contrary, our model greatly overestimates precipitation over the Sahara desert

and over the Antarctic continent, and underestimates it over oceans. In addition, the network seems to overestimate the precipitation variability in the Antarctic continent, while largely underestimating its variability at any other lovation. In fact, there is a general variability underestimation for both cloud cover and precipitation across the sphere. Finally, the cloud cover predictions are very weakly correlated with the observations (R2 < 0.6), and the precipitation predictions are not correlated to the observations at all (see Figure 12).

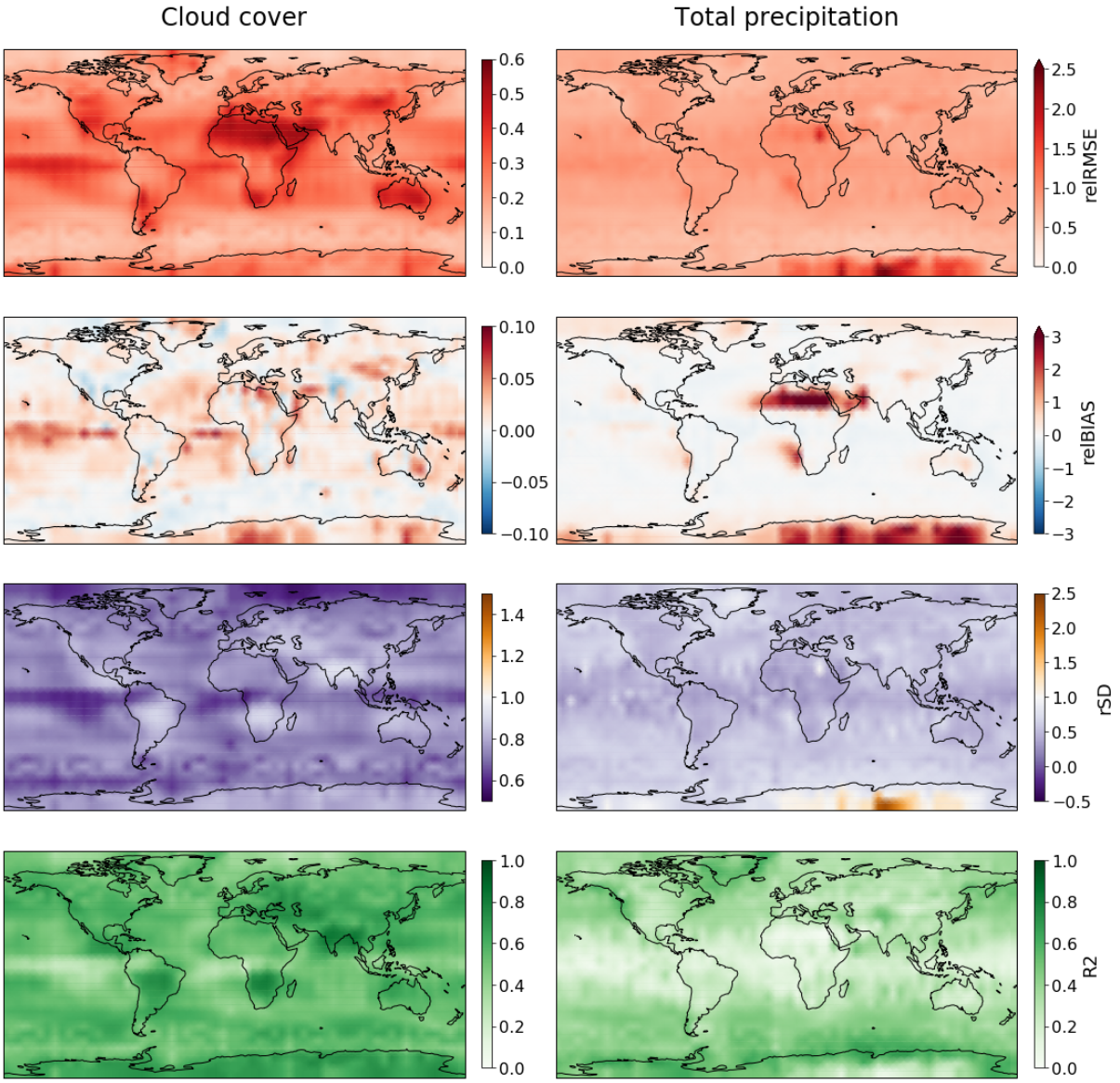


Figure 12: Spatial distribution of all skills between predictions and observations of cloud cover and total precipitation. The forecast lead time is 6 hours.

5.3 Inclusion of time sequences

This last section aims at designing an experimental setup that assesses how the using time sequences as input helps with predictability. Including time sequences in the model adds two more parameters: the length of the sequence L and the time interval between the samples in the sequence Δ_t.

We train 4 CNNs with varying L and Δ_t. The input features of these 6 CNNs are always all of the static features, Z500 and T850. Their output is always Z500 and T850. We test the CNNs predictability for all combinations of $L = \{2, 4\}$ and $\Delta_t = \{6, 12\}$.

5.3.1 Effect of sequence length

Increasing the length of the sequence L slightly helps improve the global RMSE of T850 overall. Indeed, the first T850 RMSE decreases as L increases. For T850, increasing L also helps decreasing the RMSE in the long term. This observation does not hold for Z500. Indeed, using $L > 1$ helps indeed decrease the error, but $L > 2$ does not improve the first prediction and actually makes subsequent Z500 predictions diverge faster (Figure 13). Table 2 provides a detailed comparison of RMSEs.

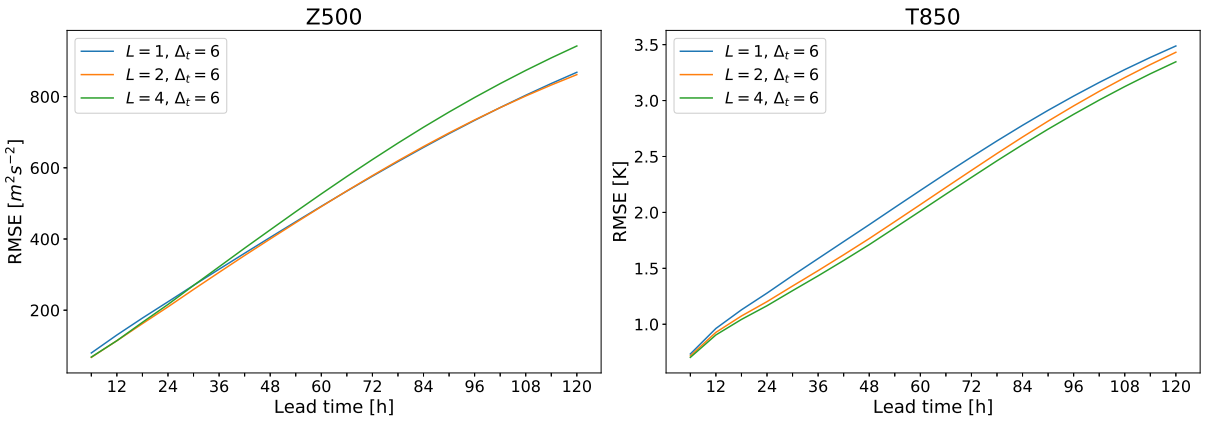


Figure 13: RMSE between the observations and the predictions for Z500 and T850, for different L, using $\Delta_t = 6$. The inclusion of clouds and precipitations degrades the predictive skill of our model for Z500 and T850 in the short term and makes predictions diverge faster.

We observe the same tendency for the $\Delta_t = 12h$ models. Although the RMSE of the first prediction of T850 slightly decreases when L increases from 2 to 4 (**1.423** and **1.412** respectively), the first Z500 RMSE slightly increases (**240.1** and **240.6** respectively). One explanation for the lack of improvement for Z500 when $L > 2$ might be that the state of the atmosphere more than 12 hours in advance does not provide additional information to the forecast, hence it is not useful for the prediction. Consequently, $L = 2$ seems to be the sequence length that most improves the predictive skill.

5.3.2 Effect of Δ_t

Increasing Δ_t from 6 to 12 decreases the RMSE slope for both Z500 and T850. Although the $\Delta_t = 12$ RMSEs start by being higher than the $\Delta_t = 6$ RMSEs, their lower slope makes the $\Delta_t = 12$ predictions more performant in the long-term. The result of this experiment is intuitive: the shorter the forecast, the more temporal resolution is needed. This is achieved by a smaller Δ_t. However, the longer the forecast, the more past temporal information is required. This is achieved by a larger Δ_t. It is interesting to note that the $\Delta_t = 6$h curves and the $\Delta_t = 12$h curves do not cross at the same forecast lead time for Z500 and T850. Z500 seems to benefit from larger Δ_t earlier than T850. This might suggest that Z500 has more temporal inertia than T850 (Figure 14).

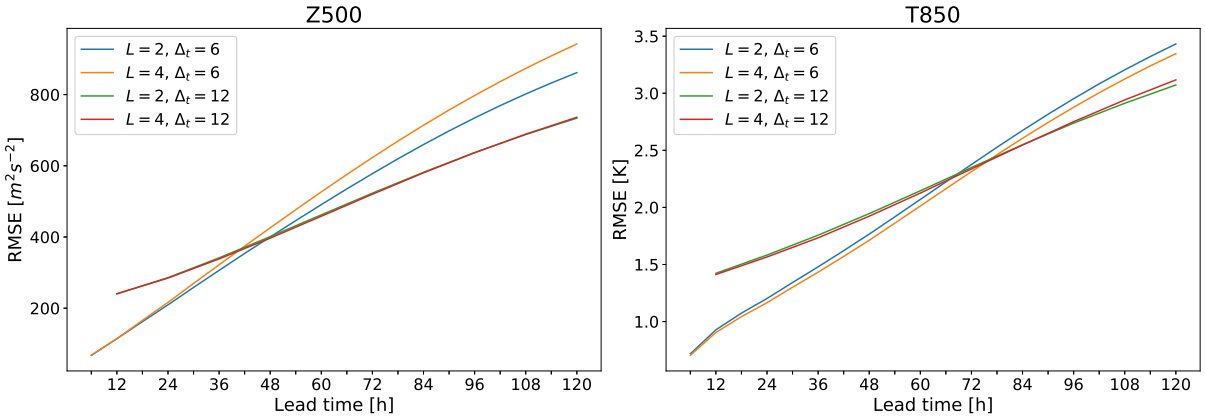


Figure 14: RMSE between the observations and the predictions for Z500 and T850, for all Δ_t and L.

The result of this experiment motivates a temporal multi-resolution prediction, in which a single model predicts short forecast lead times with high temporal resolution and long forecast lead times with smaller temporal resolution.

5.4 Best performing model

In this section we present the results we obtained with the model having the best predictive skill in the previous sections. We decided to select a model for which $\Delta_t = 6$h to allow better comparison with Rasp et al. (2020), Weyn et al. (2020) and the baselines. Table 2 presents a summary of RMSEs of our best performing models.

	All static features $L = 1$, $\Delta_t = 6$	**All static features + Z1000**	**All static features** $L = 2$, $\Delta_t = 6$	**All static features** $L = 4$, $\Delta_t = 6$
Lead time = 6h	Z500: 79.84 T850: 0.7341	Z500: 83.64 T850: 0.8296	Z500: **67.46** T850: 0.7172	Z500: 68.27 T850: **0.7039**
Lead time = 120h	Z500: 868.3 T850: 3.488	Z500: **829.0** T850: 4.714	Z500: 861.7 T850: 3.432	Z500: 942.2 T850: **3.347**

Table 2: RMSE comparison among our best performing $\Delta_t = 6$h models. The model **All static features,** $L = 2$, $\Delta_t = 6$ is the best compromise between low Z500 and T850 predictions, in both the short-term and the long-term.

The selected model is **All static features,** $L = 2$, $\Delta_t = 6$ presented in Section 5.3. We will compare our best performing model to several baselines. For easier comparison, we label our model **Ours** in figures.

5.4.1 Model benchmark

We compare our model with several baselines using the global RMSE introduced in Section 4.6.4. The baselines are persistence, global and weekly climatologies and the Operational IFS. In addition, we compare our performance to the direct CNN predictions presented in Rasp et al. (2020), and to the iterative predictions obtained by Weyn et al. (2020). Our predictions outperform the global climatology and the persistence baseline for at least 5 days. The weekly climatology is outperformed by our T850 prediction for 5 days and by our Z500 prediction for 4.5 days. Our RMSE is lower than all iterative CNN predictions and the first direct CNN prediction in Rasp et al. (2020). Although the work in Weyn et al. (2020) outperforms our model in the long-term, our Z500 prediction has a lower RMSE in the first 12 hours of forecast and our T850 prediction outperforms Weyn et al. (2020) in the first 30 hours (Figure 15).

Finally, the RMSE of our T850 at a forecast lead time of 6 hours (**0.7172**) is comparable to the RMSE of the Operational IFS (**0.6965**).

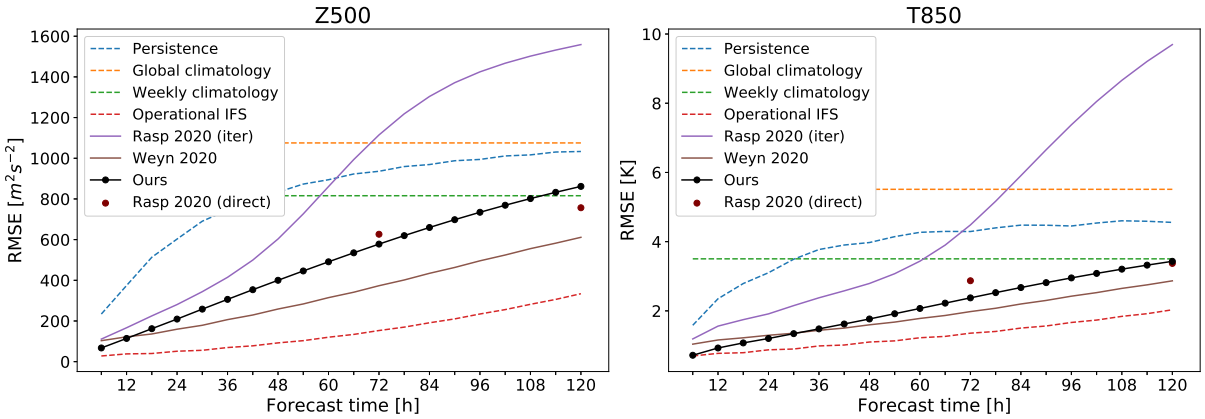


Figure 15: RMSE benchmark of our best model. Our model is compared to several baselines and previous works. Our predictions outperform climatology and persistence for 5 days, and produce state-of-the art results in early forecast lead times.

The work in Weyn et al. (2020) uses a spatial resolution of 2° and one network per face of the cube. Consequently, their CNNs use higher resolution data on smaller areas of influence. Our work uses a coarser grid and a single CNN for the entire sphere, hence making the comparison with Weyn et al. (2020) difficult. Indeed, six high resolution networks are expected to perform better than a single low resolution one. For this reason and for lack of available baselines, we omit the work in Weyn et al. (2020) from the MAE and ACC benchmarks.

The MAE curves follow the same tendencies as the RMSE curves. Our Z500 predictions perform better than persistence and global climatology for up to 5 days, and better than the weekly climatology for 4 days. In addition, our Z500 MAE curve is below the iterative CNN Rasp et al. (2020) MAE curve, although for this skill, our Z500 prediction is unable to beat the direct CNN prediction in Rasp et al. (2020). For T850 however, our model outperforms both climatologies, the persistence forecast and the direct and iterative CNN predictions in Rasp et al. (2020) for at least 5 days (Figure 16).

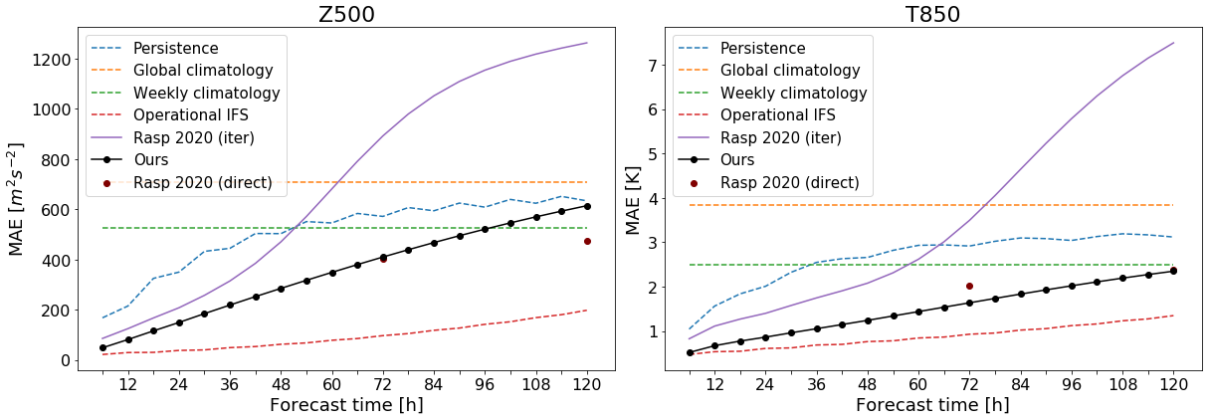


Figure 16: MAE benchmark of our best model. Using this skill, our T850 predictions beat all baselines but the Operational IFS for 5 days. Our Z500 predictions are unable to beat the direct CNN predictions in Rasp et al. (2020).

The ACC skill shows that both Z500 and T850 predictions outperform both climatologies and the persistence baseline up to 5 days, as well as the iterative CNN predictions in Rasp et al. (2020). Although our T850 predictions have a higher ACC than both direct predictions in Rasp et al. (2020), our Z500 predictions are only able to outperform the first direct prediction (Figure 17). In general, the RMSE, MAE and ACC curves provide similar information: our model performs better in the T850 prediction than in the Z500 prediction.

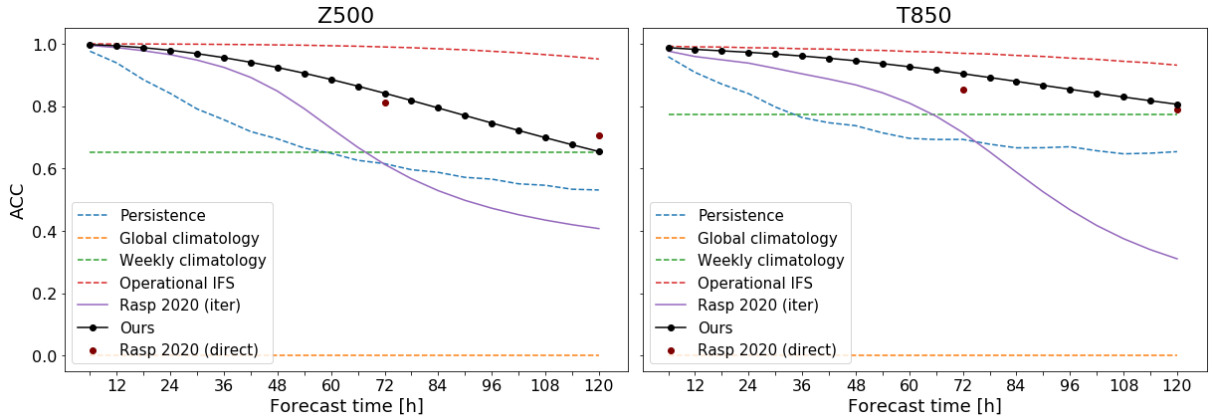


Figure 17: ACC benchmark of our best model. The predictions for both features outperform the global and weekly climatologies, the persistence forecast and the iterative CNN forecast in Rasp et al. (2020). Our Z500 forecast beats the first direct prediction from **?**, while our T850 forecast beats both.

5.4.2 Skills of our best model

This section exposes the spatial distribution of our best model for each forecast lead time and for each skill presented in Section 4.6.5. Generally, all distributions are narrow for the first forecast lead time, and widen as the forecast lead time increases. The distributions are rarely unskewed for any skill nor for any feature, meaning that the mean rarely coincides with the median. This suggests that for each skill and feature, our model shows a clear tendency. This observation is not surprising for relRMSE as they are bounded. However, the distributions of relBIAS and rSD are also generally skewed. Indeed, the data points in the relBIAS distribution tend to be negative for both Z500 and T850, indicating that our model has a tendency to underestimate the mean for most pixels. The Z500 rSD distribution is the only one resembling a normal distribution at any forecast lead time. The mean is aligned with the median, and the distribution is symmetric around it. In addition, both the mean and the median are close to 1, the ideal value, and the distribution remains fairly narrow even as the forecast lead time increases. This suggests that our model tends to correctly estimate the Z500 variability. For T850, however, the rSD distributions are skewed towards the negative values, with many outliers. Our model generally underestimates the T850 variability for all forecast lead times. Finally, the Z500 R2 skill is normally distributed and presents almost no outliers. The T850 R2 distribution has a higher mean than the one of Z500, but shows data points with low R2 skill in early forecast lead times.

The distributions of each skill vary between Z500 and T850. The T850 distributions tend to be narrower but to have more outliers (Figure 18).

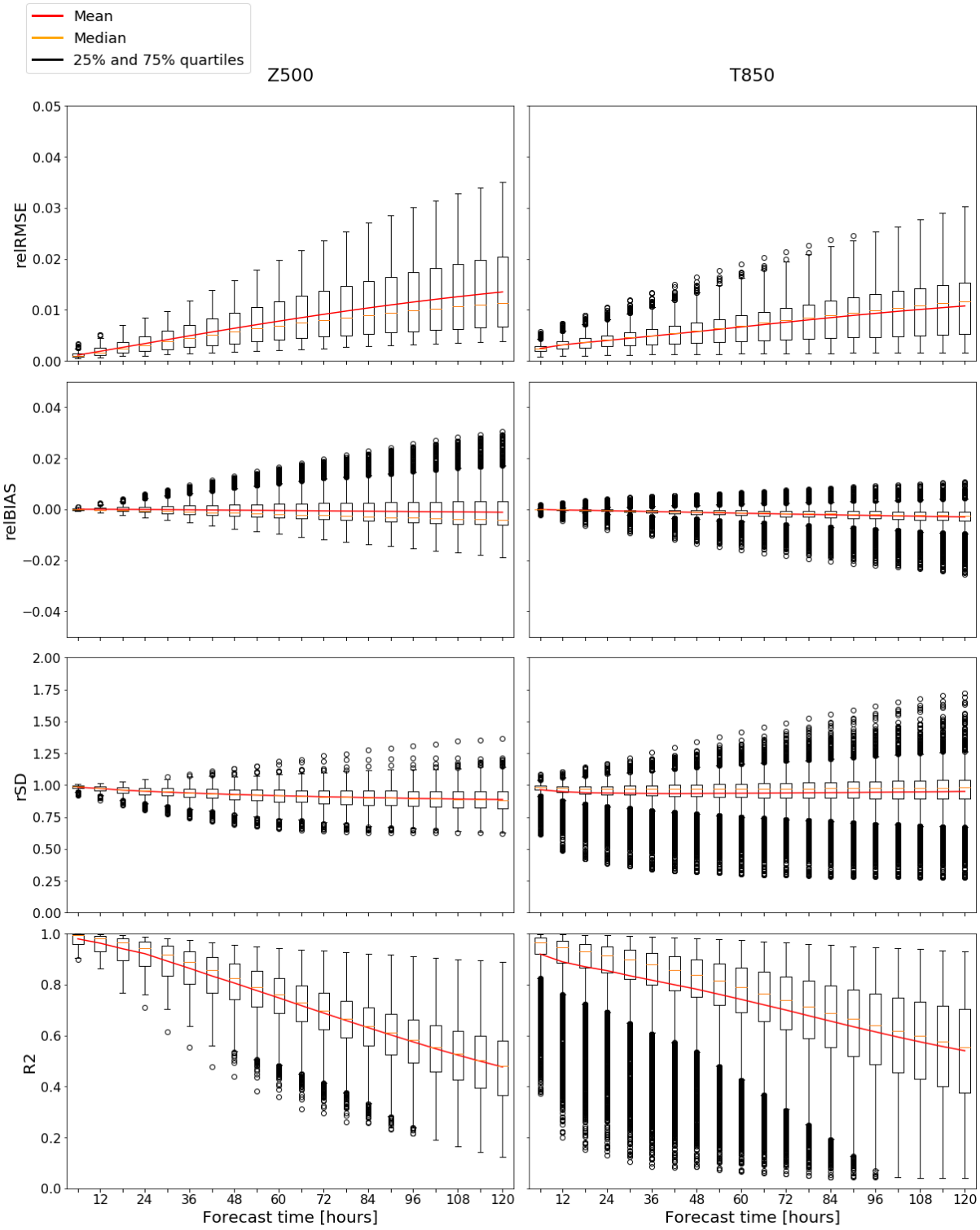


Figure 18: General skill distribution of our best model. This figure shows the 25%, 50% (median) and 75% quantiles of the distribution of each skill, as well as its mean and outliers.

We present these distributions as spatial skill maps. The metrics were computed on the HEALPix

sampling, and then interpolated onto an equiangular sampling for display. As the HEALPix pixels have the same area regardless of latitude, the display of our skill maps using a planar projection distorts the area of the errors, over-representing the poles and under-representing the equator.

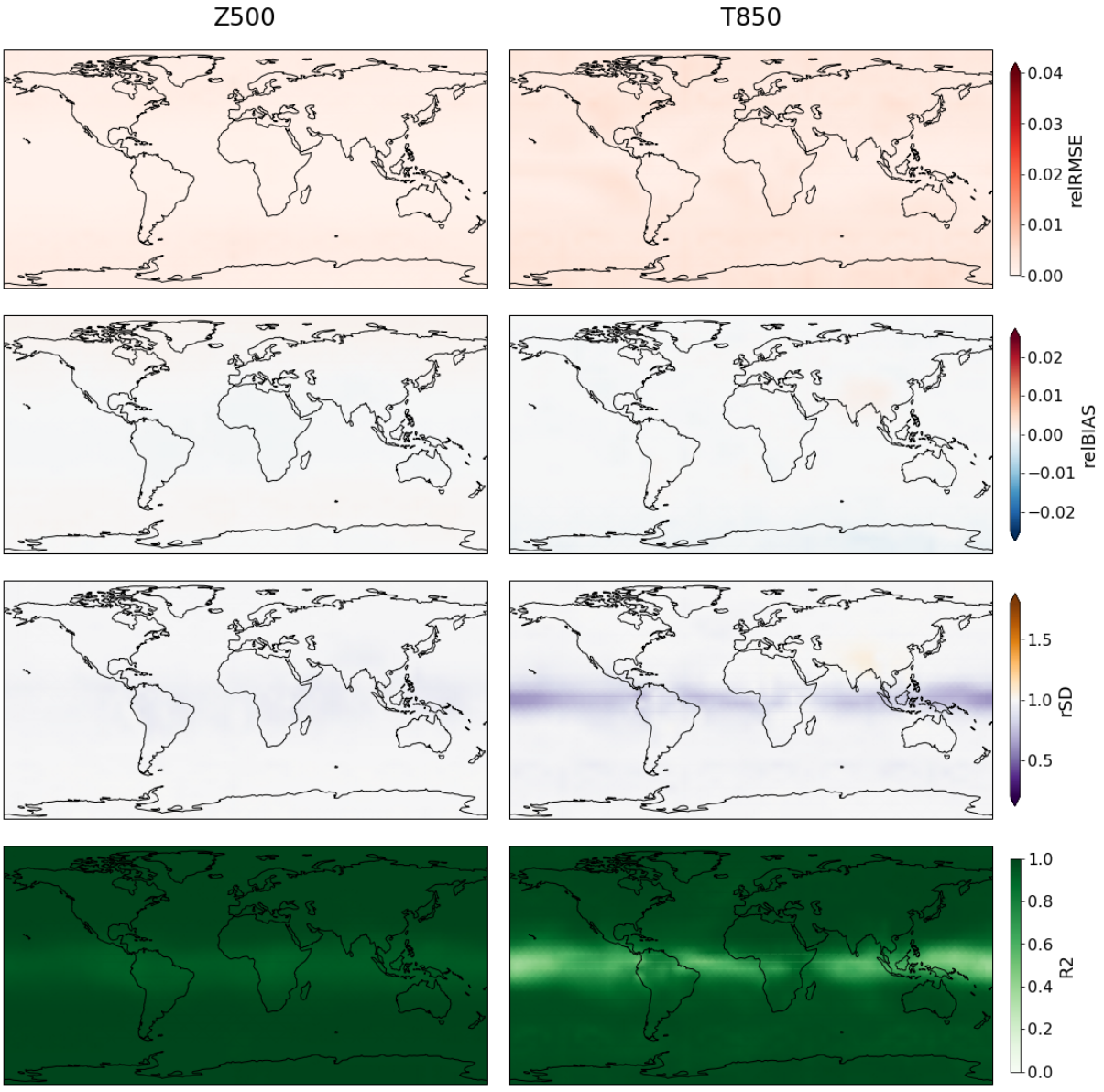


Figure 19: Spatial skill distribution of our best model at the first prediction (forecast lead time = 6h).

For our first prediction with a forecast lead time of 6 hours, the magnitude of the error is low, with not a clear over/under estimation tendency. The variability is also generally well reproduced, except along the tropics for what concerns T850, where it is underestimated. In such tropical band, the correlation between predictions and observations is reduced (see Figure 19). We hypothesize that such difficulties in reproducing the T850 dynamics at the tropics might be affected by complex physical processes, such as convection, that cannot be captured at our spatial or temporal resolutions. These processes cause large amounts of water vapour to condense and latent heat release, which strongly affects temperature. Since

this is the first prediction, this artifact is not caused by iterations.

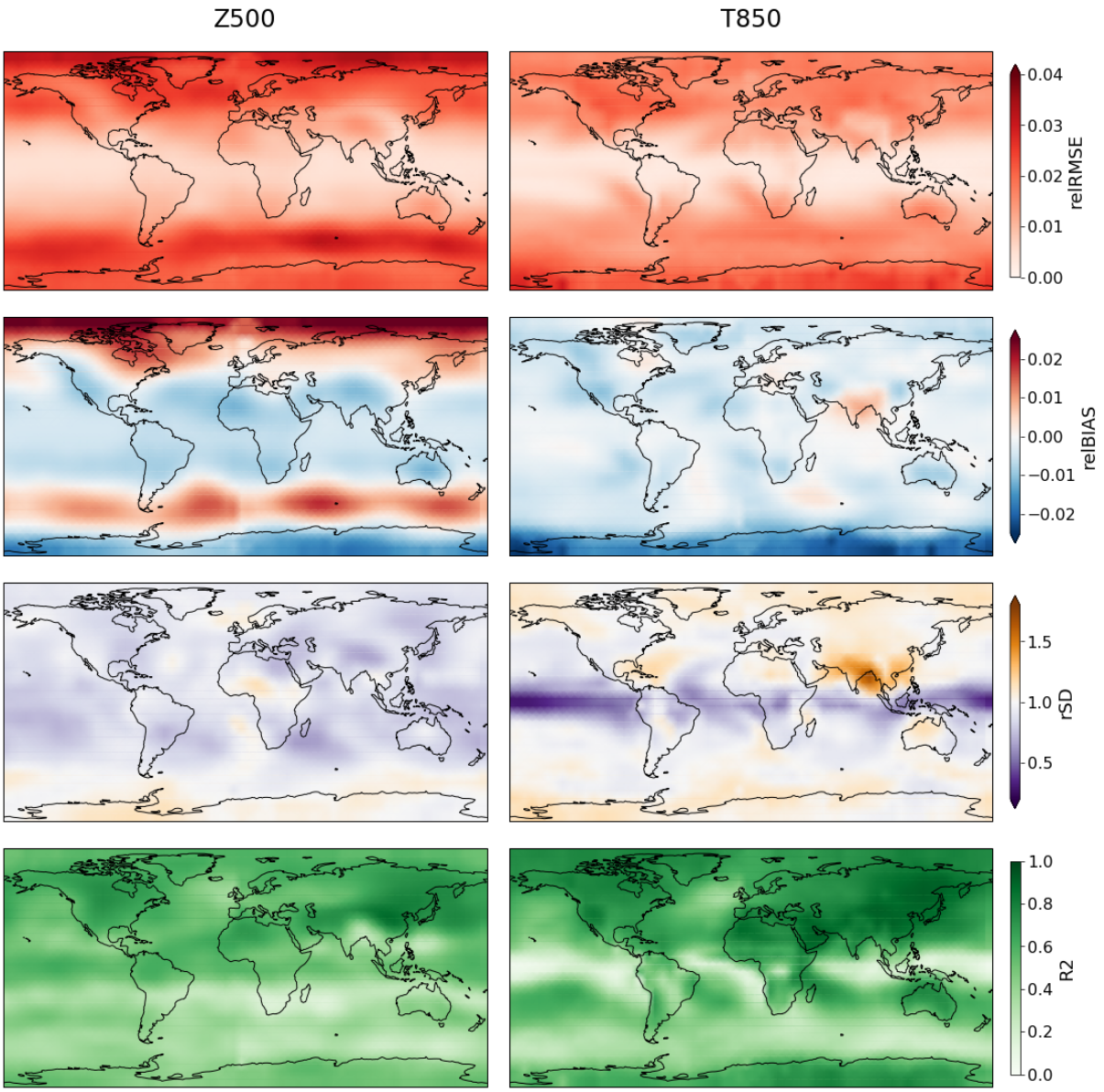


Figure 20: Spatial skill distribution of our best model at the last prediction (forecast lead time = 120h).

Our last prediction, at a forecast lead time of 120 hours, follow the tendencies observed in the first prediction. The error magnitude has remained stable in the tropical regions, and has increased with latitude, especially for Z500. For T850, relBIAS values are generally negative, except for the Indian sub-continent. In that region, the T850 bias and the T850 are overestimated. For Z500, the bias is largely overestimated in regions close to the North pole and in the Antarctic Ocean, while being underestimated in the other locations. The Z500 variability is in general well estimated, although slightly overestimated over the Antarctic continent and slightly underestimated in every other location. For both Z500 and T850, the predictions are in general weakly correlated with the observations. Finally, the artifact observed in the first T850 prediction is reinforced. Indeed, our model underestimates the tropical T850 variability by up to 100% in the Pacific. In those regions, the T850 correlation between predictions and observations is

close to 0 (see Figure 20).

The results obtained at the poles are partly explained by the error accumulation with the iterations. Another source of error in such regions might also be ERA5 itself, as those regions can have unreliable measurements.

5.4.3 Climatology reproduction

Our best model is able to reproduce the T850 monthly latitude climatologies, only deviating from observations in magnitude and not in trend. In general, our model underestimates T850 climatologies, especially in the Antarctic. For Z500, the tendency observed in the spatial skill distributions is confirmed. The Z500 monthly climatologies are overestimated in northern regions, as well as in the Southern Ocean, while slightly underestimated in mid-latitudes, tropical regions and the Antarctic continent (see Figure 21).

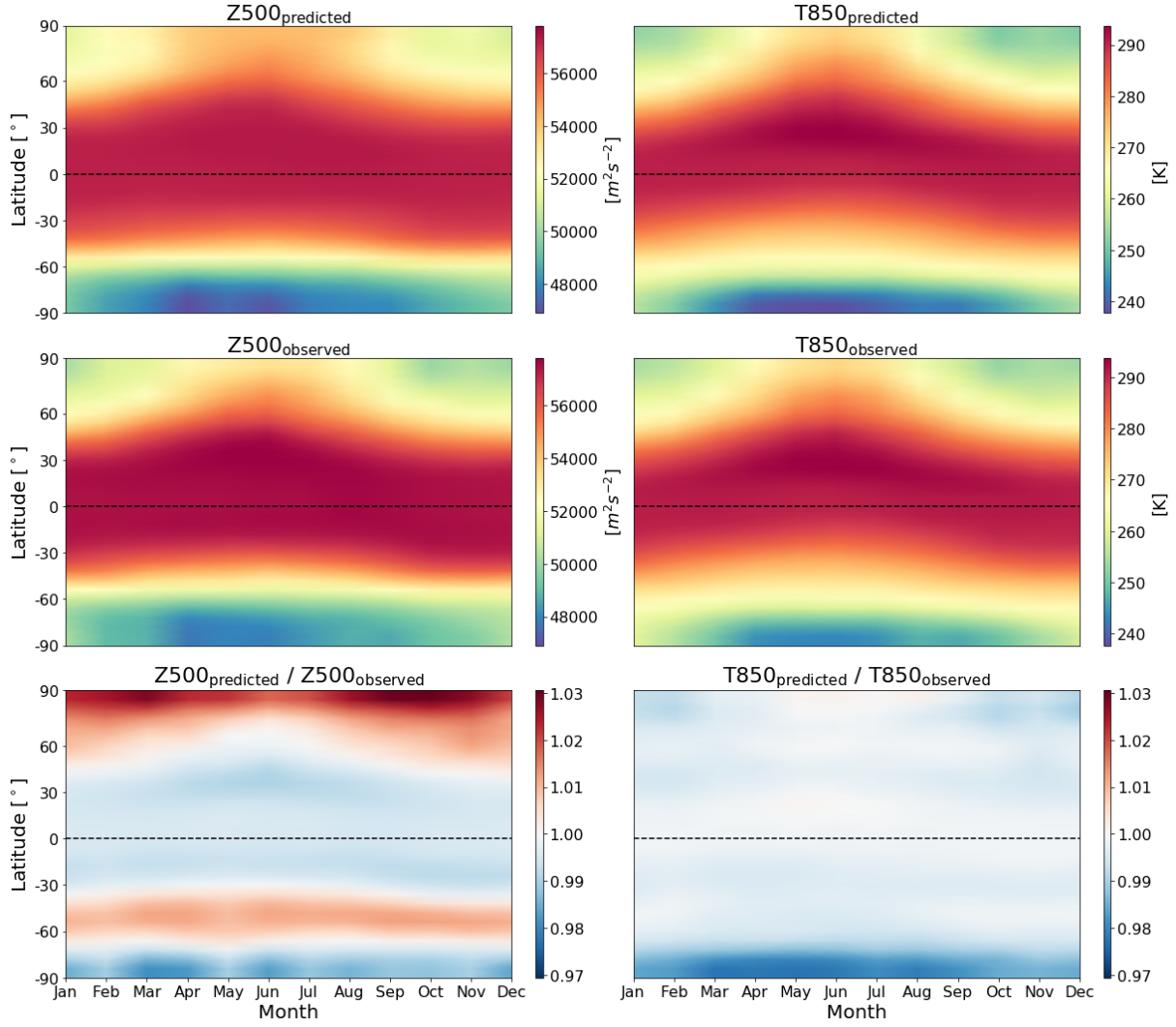


Figure 21: Hovmöller diagram of the monthly mean for Z500 and T850 as a function of latitude at the last prediction (forecast lead time = 120h).

6 Conclusions

This research provides a data-driven method for numerical medium-range weather forecasting that respects the spatio-temporal geometry of the problem. We present a CNN combining spherical graph convolutions with temporal sequences. The network is trained in an iterative manner to constrain the network to output stable long-term forecasts. We test the influence of static and dynamic atmospheric features, time sequences and temporal discretization on the predictive skill. The performance is evaluated on the features Z500 and T850 using a wide range of global and spatially distributed metrics and benchmarked to several baselines and previous works.

Experimental results show that predictions benefit in the short-term from the addition of static features. Nevertheless, as the forecast lead time increases, some static features make the predictions diverge faster. While errors accumulate on the predicted features due to iterations, static features remain unchanged. The relationship learnt by the network between static and dynamic features is not guaranteed to hold when the predictions greatly differ from training data. As such, including static features in late iterations might induce more error.

When all static features are maintained, the inclusion of the 1000-hPa geopotential (Z1000) slightly reduces the RMSE divergence in Z500 as the forecast lead time increases. Nevertheless, it is not the case for T850. While the geopotential height mostly depends on the density and temperature of the air situated underneath the considered pressure level, the temperature forecast might not benefit from such information. Replacing Z1000 by the geopotential at 850 hPa may improve both Z500 and T850 predictions. Alternatively, these results could be due to a limitation in capacity of our network, which might prevent it from correctly modeling three fields. Nevertheless, the improvement of the Z500 forecast supports the integration of a vertical dimension to the space-time representation of the data to account for the atmosphere thickness.

Increasing the length of the input time sequence L showed to have little effect on predictability. For T850, we observed a slight improvement as L increased. This was not reproduced for Z500, for which the improvement stopped for a sequence length of two. We deduce that the additional previous time steps are not sufficiently correlated with the prediction, hence the additional information is not useful for medium-range Z500 forecasts. Lowering the temporal resolution increased the prediction error for short forecast lead times but further stabilized the prediction divergence in the long-term.

The encoder/decoder framework paired with skip connections capture coarse spatial patterns in extended areas while maintaining local spatial detail. This makes the Unet a spatial multi-resolution architecture. Our experiments involving time sequences confirm that adding the temporal dimension to the multi-resolution approach is imperative for accurate forecasts in both short and long terms. As such, shorter forecasts benefit from high spatio-temporal precision on a small area of influence, while longer forecasts profit from coarser but more extended time and space information.

Our best model consists in a sequence of two time steps including all static variables, Z500, and T850. Our predictions outperform the global climatology and the persistence baseline for at least 5 days, and the weekly climatology for more than 4 days. Our Z500 predictions surpass the iterative CNN and the first direct CNN predictions in Rasp et al. (2020). Finally, our model produces stat-of-the-art T850 predictions, which outperform all low resolution IFS baselines as well as all previous works for the first 30 forecast hours.

References

Bauer, P., Thorpe, A., and Brunet, G. (2015). The quiet revolution of numerical weather prediction. *Nature*, 525:47–55.

Bengtsson, L. (1985). Medium-range forecasting at the ECMWF. In Saltzman, B., editor, *Issues in Atmospheric and Oceanic Modeling*, volume 28 of *Advances in Geophysics*, pages 3 – 54. Elsevier.

Bjerknes, V. (1904). Weather prediction and the prospect for its improvement.

Bubnová, R., Hello, G., Bénard, P., and Geleyn, J.-F. (1995). Integration of the fully elastic equations cast in the hydrostatic pressure terrain-following coordinate in the framework of the ARPEGE/Aladin NWP System. *Monthly Weather Review*, 123(2):515–535.

Buizza, R. and Leutbecher, M. (2015). The forecast skill horizon. *Quarterly Journal of the Royal Meteorological Society*, 141(693):3366–3382.

Businger, J. A., Wyngaard, J. C., Izumi, Y., and Bradley, E. F. (1971). Flux-Profile Relationships in the Atmospheric Surface Layer. *Journal of Atmospheric Sciences*, 28(2):181–189.

C3S, Copernicus Climate Change Service. (2017). ERA5: Fifth generation of ECMWF atmospheric reanalyses of the global climate.

Charney, J. G., Fjørtoft, R., and Neumann, J. V. (1950). Numerical integration of the barotropic vorticity equation. *Tellus*, 2(4):237–254.

Cohen, T., Geiger, M., Köhler, J., and Welling, M. (2018). Spherical CNNs.

Coiffier, J. (2011). *Fundamentals of Numerical Weather Prediction*. Cambridge University Press.

Courant, R., Friedrichs, K., and Lewy, H. (1928). Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen*, 100:32–74.

Csáji, B. C. (2001). Approximation with Artificial Neural Networks. Master's thesis, Eötvös Loránd University (ELTE), Budapest, Hungary.

Deardorff, J. W. (1972). Numerical Investigation of Neutral and Unstable Planetary Boundary Layers. *Journal of Atmospheric Sciences*, 29(1):91–115.

Deardorff, J. W. (1977). Efficient prediction of ground surface temperature and moisture, with inclusion of a layer of vegetation.

Defferrard, M. (2020). Personal conversation.

Defferrard, M., Milani, M., Gusset, F., and Perraudin, N. (2020). DeepSphere: a graph-based spherical CNN. In *International Conference on Learning Representations*.

Defferrard, M., Perraudin, N., Kacprzak, T., and Sgier, R. (2019). DeepSphere: towards an equivariant graph-based spherical CNN. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

Driscoll, J. and Healy, D. (1994). Computing fourier transforms and convolutions on the 2-sphere. *Advances in Applied Mathematics*, 15(2):202 – 250.

Düben, P. and Bauer, P. (2018). Challenges and design choices for global weather and climate models based on machine learning. *Geoscientific Model Development*, 11:3999–4009.

Esteves, C., Allen-Blanchette, C., Makadia, A., and Daniilidis, K. (2019). Learning SO(3) equivariant representations with spherical CNNs. *International Journal of Computer Vision*.

Feldmann, M. (2020). Personal conversation.

Ghiggi, G., Humphrey, V., Seneviratne, S. I., and Gudmundsson, L. (2019). GRUN: an observation-based global gridded runoff dataset from 1902 to 2014. *Earth System Science Data*, 11(4):1655–1674.

Glorot, X., Bordes, A., and Bengio, Y. (2011). Deep sparse rectifier neural networks. In Gordon, G., Dunson, D., and Dudík, M., editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 315–323, Fort Lauderdale, FL, USA. PMLR.

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

Gorski, K. M., Hivon, E., Banday, A. J., Wandelt, B. D., Hansen, F. K., Reinecke, M., and Bartelmann, M. (2005). HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere. *The Astrophysical Journal*, 622(2):759–771.

Gorski, K. M., Wandelt, B. D., Hansen, F. K., Hivon, E., and Banday, A. J. (1999). The HEALPix primer.

Hennermann, K. (2020). ERA5: data documentation.

Hersbach, H., Bell, B., Berrisford, P., Hirahara, S., Horányi, A., Muñoz-Sabater, J., Nicolas, J., Peubey, C., Radu, R., Schepers, D., Simmons, A., Soci, C., Abdalla, S., Abellan, X., Balsamo, G., Bechtold, P., Biavati, G., Bidlot, J., Bonavita, M., De Chiara, G., Dahlgren, P., Dee, D., Diamantakis, M., Dragani, R., Flemming, J., Forbes, R., Fuentes, M., Geer, A., Haimberger, L., Healy, S., Hogan, R. J., Hólm, E., Janisková, M., Keeley, S., Laloyaux, P., Lopez, P., Lupu, C., Radnoti, G., de Rosnay, P., Rozum, I., Vamborg, F., Villaume, S., and Thépaut, J.-N. (2020). The ERA5 Global Reanalysis. *Quarterly Journal of the Royal Meteorological Society*.

Hersbach, H. and Dee, D. (2016). ERA5 reanalysis is in production. In *ECMWF Newsletter*, volume 147. EMS.

Hyndman, R. and Athanasopoulos, G. (2018). *Forecasting: Principles and Practice*. OTexts, Australia, 2nd edition.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, page 448–456. JMLR.org.

James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. (2009). What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th International Conference on Computer Vision*, pages 2146–2153.

Jiang, C., Huang, J., Kashinath, K., Prabhat, M., Marcus, P., and Niessner, M. (2019). Spherical CNNs on unstructured grids.

Katayama, A. (1972). A simplified scheme for computing radiative transfer in the troposphere. Technical report, UCLA, Department of Meteorology.

Laprise, R. (1992). The euler equations of motion with hydrostatic pressure as an independent variable. *Monthly Weather Review*, 120(1):197–207.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature Cell Biology*, 521(7553):436–444.

Lorenz, E. N. (1969). The predictability of a flow which possesses many scales of motion. *Tellus*, 21(3):289–307.

Louis, J.-F. (1979). A parametric model of vertical eddy fluxes in the atmosphere. *Boundary-Layer Meteorology*, 17(2):187–202.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. The MIT Press.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA. Omnipress.

Palmer, T., Brankovic, C., Molteni, F., and Mureau, R. (1989). Predictability in the medium range and beyond. In *Seminar on 10 years of Medium-range Weather Forecasting, 4-8 September 1989*, volume I, pages 129–158, Shinfield Park, Reading. ECMWF, ECMWF.

Platzman, G. W. (1979). The ENIAC computations of 1950 — gateway to numerical weather prediction. *Bulletin of the American Meteorological Society*, 60(4):302–312.

Rasp, S., Düben, P., Scher, S., Weyn, J., Mouatadid, S., and Thuerey, N. (2020). WeatherBench: A benchmark dataset for data-driven weather forecasting.

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

Richardson, L. F. (1922). *Weather prediction by numerical process*. Cambridge University Press.

Rodgers, C. and Walshaw, C. (1967). The computation of infra-red cooling rate in planetary atmospheres. 93:555–556.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Navab, N., Hornegger, J., Wells, W. M., and Frangi, A. F., editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham. Springer International Publishing.

Scher, S. (2018). Toward data-driven weather and climate forecasting: Approximating a simple general circulation model with deep learning. *Geophysical Research Letters*, 45.

Scher, S. and Messori, G. (2019). Weather and climate forecasting with neural networks: using general circulation models (GCMs) with different complexity as a study ground. *Geoscientific Model Development*, 12:2797–2809.

Shuman, D. I., Narang, S. K., Frossard, P., Ortega, A., and Vandergheynst, P. (2013). The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. *IEEE Signal Processing Magazine*, 30(3):83–98.

Smagorinsky, J. (1962).

Talley, L. D., Pickard, G. L., Emery, W. J., and Swift, J. H. (2011). Chapter 7 - Dynamical processes for descriptive ocean circulation. In Talley, L. D., Pickard, G. L., Emery, W. J., and Swift, J. H., editors, *Descriptive Physical Oceanography (Sixth Edition)*, pages 187 – 221. Academic Press, Boston, sixth edition edition.

Tanguay, M., Robert, A., and Laprise, R. (1990). A semi-implicit send-lagrangian fully compressible regional forecast model. *Monthly Weather Review*, 118(10):1970–1980.

Thorpe, A. (2012). European Centre for Medium-Range Weather Forecasts: Interview published in "International Innovation".

Wallace, J. M. and Hobbs, P. V. (2006). *Atmospheric Science: An Introductory Survey*. Elsevier, 2 edition.

Warner, T. T. (2010). *Numerical Weather and Climate Prediction*. Cambridge University Press.

Weyn, J., Durran, D., and Caruana, R. (2020). Improving data-driven global weather prediction using deep convolutional neural networks on a cubed sphere.

Weyn, J., Durran, D. R., and Caruana, R. (2019). Can machines learn to predict weather? Using deep learning to predict gridded 500-hpa geopotential height from historical weather data. *Journal of Advances in Modeling Earth Systems*, 11(8):2680–2693.

Xu, B., Wang, N., Chen, T., and Li, M. (2015). Empirical evaluation of rectified activations in convolutional network.