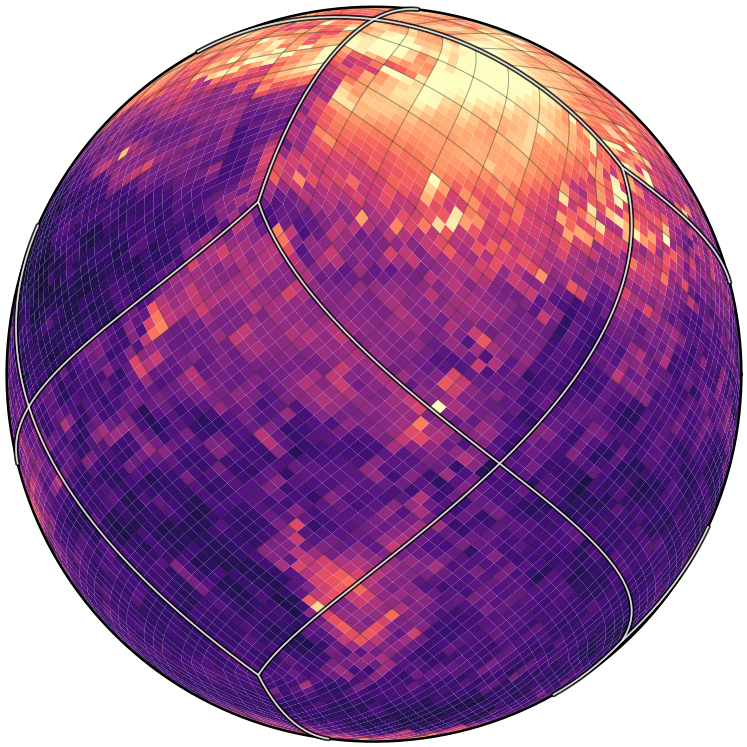

Deep Learning Climate Model Emulation with PEAR

Adapting a Transformer-Based Weather Forecasting Model on the HEALPix Grid to the Climate Domain

Master's thesis in Complex Adaptive Systems

TAGE TYKESSON

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2026
www.chalmers.se

Master's thesis 2026

Deep Learning Climate Model Emulation with PEAR

Adapting a Transformer-Based Weather Forecasting Model on the HEALPix Grid to the Climate Domain

TAGE TYKESSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Chalmers University of Technology
Gothenburg, Sweden 2026

Deep Learning Climate Model Emulation with PEAR
Adapting a Transformer-Based Weather Forecasting Model on the HEALPix Grid to the Climate Domain
Tage Tykesson

Supervisor: Daniel Persson, Department of Mathematical Sciences
Examiner: Daniel Persson, Department of Mathematical Sciences

Master's Thesis 2026
Department of Mathematical Sciences
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Equal area HEALPix gridding of a timestep prediction residual map for the Climate PEAR model.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2026

Deep Learning Climate Model Emulation with PEAR
Adapting a Transformer-Based Weather Forecasting Model on the HEALPix Grid to the Climate Domain

Tage Tykesson
Department of Mathematical Sciences
Chalmers University of Technology

Abstract

Climate model emulation aims to approximate the forcing–response behaviour of computationally expensive Earth system models at a fraction of their original cost. This thesis adapts PEAR, a HEALPix-based Volumetric Swin Transformer originally developed for medium-range weather prediction on the sphere, to the task of climate model emulation. Instead of predicting future atmospheric states from initial conditions, the adapted model learns to map greenhouse-gas and aerosol emission forcings to global fields of near-surface air temperature and precipitation. The model is evaluated on ClimateSet, a machine-learning-ready dataset containing processed CMIP6 outputs from multiple Earth system models, and benchmarked against UNet and ClimaX baselines from the ClimateSet benchmarks. Across 15 climate models and 5 random seeds, PEAR achieves competitive performance despite its small size, with 4.3M trainable parameters compared with 18.3M for UNet and 106.6M for ClimaX. Although the larger ClimaX model obtains the strongest overall results, PEAR performs favourably compared with UNet, outperforming it on 13 of the 15 climate models in overall RMSE on normalized labels. The strongest results are found for precipitation, where PEAR achieves the lowest RMSE on a majority of the evaluated models. These results suggest that PEAR provides a promising parameter-efficient architecture for climate model emulation, with further potential for improvement through architectural, temporal, and capacity-related refinements.

Keywords: Climate Model Emulation, PEAR, Machine Learning, Deep Learning, ClimateSet, Vision Transformer, Swin Transformer, HEALPix.

Acknowledgements

I want to thank Daniel Persson, Hampus Linander and Jan Gerken, for their wisdom, expertise, and generosity with their time throughout this project. Their constructive feedback and insightful guidance have been invaluable to the work, as has the welcoming research environment they have provided. Throughout the project, I have always felt encouraged to bring my own ideas forward, to ask questions freely, and to think independently, all of which has allowed me to grow as a researcher. For all of this, I am sincerely grateful.

I also want to thank all the people who have supported me along the way: my friends and my family. You have been there for me when I have needed it the most, and without you this journey would have been much harder. To all of you who have been there through both the highs and the lows: a huge thank you.

Tage Tykesson, Gothenburg, 2026

Contents

1

Introduction

Climate model emulation has recently attracted increasing attention from the machine learning community, with the goal of approximating the forcing–response behaviour of large Earth system models at a fraction of their computational cost. Individual climate simulations often require weeks or months to run, severely limiting the number and diversity of scenarios that can be explored and making it difficult to systematically investigate scientific and policy-relevant questions [1]. By learning from existing simulations, emulators act as statistical surrogates that approximate complex model behaviour orders of magnitude faster, enabling large ensembles, scenario exploration, uncertainty decomposition, and experiments that would otherwise be computationally infeasible [2]. Current efforts have evolved from simple linear and Gaussian process methods to advanced deep learning architectures capable of capturing high-dimensional, nonlinear relationships across Earth system components [2, 3].

This thesis investigates whether PEAR, a HEALPix-based Swin Transformer originally developed for medium-range weather prediction, can be adapted to the task of climate model emulation. PEAR builds on Pangu-Weather [4] by adapting it to operate natively on the HEALPix grid, in which each pixel covers the same surface area, avoiding the unphysical pole-clustering that affects equiangular discretizations of the sphere. The resulting architecture achieves strong forecasting performance in its native weather forecasting setting [5], motivating exploration of whether such a spherical architecture can achieve competitive results on other Earth system tasks such as climate model emulation.

This thesis pursues that direction using ClimateSet, a large-scale, machine-learning-ready dataset designed to support climate model emulation across a diverse set of Earth system models [6]. ClimateSet contains inputs and outputs from 36 CMIP6 models; in this work the publicly available subset of 21 models is used. ClimateSet provides monthly forcing inputs from Input4MIPs for four primary climate forcing agents—CO_2, SO_2, CH_4, and black carbon—spanning one historical scenario and four future Shared Socioeconomic Pathways, with target outputs of monthly averaged near-surface air temperature and precipitation fields.

The aim of this thesis is to adapt the PEAR architecture to climate model emulation and evaluate its performance in this new setting. This involves modifying the model

for the ClimateSet input–output structure, training it under the ClimateSet experimental setup, and comparing its performance against established machine learning baselines. A secondary goal is to investigate whether temporal information can be incorporated into PEAR by extending the model from a single-timestep formulation to a multi-timestep formulation to improve climate emulation performance.

The adapted PEAR model achieves competitive performance on the ClimateSet emulation task while using substantially fewer trainable parameters than the main baseline models. PEAR performs favourably compared with the UNet baseline despite being considerably smaller, outperforming it on 13 of the 15 evaluated climate models in overall RMSE, though ClimaX obtains the strongest overall results. Performance differs between target variables: ClimaX performs strongest on near-surface air temperature, whereas PEAR achieves the strongest precipitation performance, obtaining the lowest RMSE on 8 of the 15 evaluated climate models. These results suggest that PEAR is a promising parameter-efficient architecture for climate model emulation, with further gains potentially possible through additional architectural refinements.

Part of the results in this thesis are included in the following preprint:

PEAR: Equal Area Weather Forecasting on the Sphere

H. Linander, T. Tykesson, P. Rosso, C. Petersson, D. Persson, and J. E. Gerken, arXiv preprint arXiv:2505.17720v3, 2025.

2

Background

2.1 Climate modeling

Climate models are fundamental tools used to understand and project the evolution of the Earth's climate system. As the climate affects ecosystems, infrastructure, agriculture, economies, and human societies worldwide, understanding how it may change in the future is an important scientific and societal challenge. Changes in temperature, precipitation patterns, sea level, and the frequency of extreme weather events can have far-reaching consequences for food production, water resources, biodiversity, public health, and critical infrastructure. Reliable climate projections are therefore important not only for improving scientific understanding of the climate system, but also for long-term planning, climate adaptation, and risk management at both regional and global scales [7].

The climate system is commonly divided into five interacting components: the atmosphere, hydrosphere, cryosphere, land surface, and biosphere. These components exchange energy, moisture, momentum, and chemical constituents through a large number of coupled physical, chemical, and biological processes. Human activity is often treated as an external forcing on the climate system through emissions and land-use changes. Because of the strong coupling between components, changes in one part of the system can propagate and amplify through feedback mechanisms operating across multiple spatial and temporal scales [8].

Modern climate simulations are primarily performed using Global Climate Models (GCMs) and Earth System Models (ESMs). Traditional GCMs mainly represent the physical dynamics of the atmosphere and oceans through numerical approximations of the governing equations of fluid dynamics, thermodynamics, and radiative transfer. Earth System Models extend these frameworks by additionally incorporating components such as atmospheric chemistry, the carbon cycle, vegetation dynamics, and ocean biogeochemistry, allowing for more comprehensive representations of climate feedbacks and long-term Earth system interactions [9].

Despite being grounded in physical laws, climate models contain uncertainties arising from finite spatial resolution, incomplete knowledge of physical processes, parameterizations of unresolved phenomena such as clouds and convection, and differences in numerical implementations. In addition, the climate system itself exhibits sub-

stantial internal variability, where complex nonlinear interactions between components of the climate system can produce natural fluctuations even under identical external forcing conditions. Examples of such variability include large-scale oscillatory patterns such as the El Niño–Southern Oscillation (ENSO), which can strongly influence global temperature and precipitation patterns across multiple timescales. As a result, climate simulations may evolve differently even when initialized under very similar conditions.

Consequently, many different ESMs exist, each making slightly different assumptions and approximations. Running multiple climate models, often together with multiple initial-condition realizations of the same model, therefore provides important information about both model uncertainty and the range of possible climate responses arising from internal variability. Rather than representing a weakness of climate science, this diversity of models is valuable, as multi-model ensembles provide a way to quantify uncertainty, evaluate robustness across simulations, and better understand the sensitivity of the climate system to different processes and forcing scenarios [1]. This motivates coordinated intercomparison projects in which multiple models are evaluated under standardized experimental conditions.

2.2 CMIP

The Coupled Model Intercomparison Project (CMIP) is the main international framework for coordinated climate model intercomparison. Coordinated by the World Climate Research Programme (WCRP), CMIP defines standardized experiments that allow ESMs developed by different modelling centres to be evaluated under common simulation settings [10, 11]. Since its first phase in the mid-1990s, CMIP has become central to climate model evaluation and provides an important scientific basis for the assessment reports of the Intergovernmental Panel on Climate Change (IPCC).

In CMIP experiments, modelling centres run their own ESMs using prescribed experimental configurations and external forcings. These forcings include greenhouse gas concentrations, aerosol emissions, land-use changes, volcanic forcing, and solar variability [11]. The resulting simulations are distributed as long-term gridded datasets containing climate variables such as surface temperature, precipitation, atmospheric and oceanic fields, and sea ice. Because the experimental protocols are standardized, differences between model outputs can be analysed in relation to differences in model formulation, resolution, parameterizations, and internal variability, rather than arising from entirely incompatible experimental setups.

CMIP6, introduced in 2016, expanded the scope of previous CMIP phases through a larger set of coordinated experiments and future forcing scenarios [11]. A central component of CMIP6 is the use of Shared Socioeconomic Pathways (SSPs), which describe possible future trajectories of socioeconomic development and associated emissions [12]. The SSPs encode assumptions about factors such as population growth, economic development, land use, air pollution, energy demand, and the

transition from fossil fuels to low-carbon energy systems. Importantly, the SSPs do not prescribe the climate response itself. Instead, they define external forcing pathways that are used as inputs to ESMs, while the resulting changes in temperature, precipitation, circulation, and other climate variables depend on the dynamics and climate sensitivity of each model.

Common CMIP6 future scenarios include SSP1-2.6, SSP2-4.5, SSP3-7.0, and SSP5-8.5. The first part of the name denotes the socioeconomic pathway, while the second part approximately indicates the resulting radiative forcing level in 2100, measured in W/m^2. SSP1-2.6 represents a low-emission pathway with strong mitigation and a transition toward sustainability. SSP2-4.5 represents an intermediate pathway in which socioeconomic development broadly follows historical trends. SSP3-7.0 corresponds to a high-emission pathway characterized by regional rivalry, slow development, and limited climate policy. SSP5-8.5 describes a fossil-fuel-intensive pathway with very high emissions. Together, these scenarios allow CMIP6 models to be evaluated across a range of possible future forcing conditions, from strong mitigation to high-emission futures.

2.3 Climate model emulation

Running a climate model for a given set of inputs is computationally very expensive, with individual simulations often requiring weeks or months to run. This severely limits both the number and diversity of scenarios that can be explored, making it difficult to systematically investigate many policy-relevant and scientific questions using traditional simulations alone [1]. Machine learning-based emulators address this limitation by acting as statistical surrogates that approximate the behavior of complex climate simulators at orders-of-magnitude lower computational cost. By producing rapid projections, emulators make it possible to perform thousands of experiments, interpolate between policy scenarios, and decompose sources of uncertainty arising from model parameters, internal variability, or forcings, which would otherwise be impractical to investigate systematically.

Current efforts in the field have evolved from simple linear approximations, Gaussian processes and stochastic methods, to advanced deep learning architectures capable of capturing high-dimensional, nonlinear relationships across various Earth system components [2] [3]. Specialized emulators are being developed for both ESMs as well as for specific sub-components of the climate system. Furthermore, there is a significant push to establish shared, ML-ready benchmarks and to transition emulators from research prototypes into modular, reliable software with standardized interfaces and documentation, making these powerful tools interactive and accessible to the broader scientific community [2]. Examples of such machine-learning-ready datasets for climate model emulation are ClimateBench [13], ClimateLearn [14], ClimART [15] and ClimateSet [6].

2.4 Gridding - HEALPix

Numerically solving equations or representing data on the sphere requires discretizing the spherical surface. A common choice in climate and weather modelling is the latitude–longitude grid, which corresponds to a spherical curvilinear coordinate system. This grid is simple and regular in angular coordinates, but it is not uniform in physical space. For an equiangular latitude–longitude grid, the meridional spacing is approximately constant, while the zonal distance between neighbouring grid points decreases with latitude as $\cos\phi$, where ϕ denotes latitude. Consequently, grid cells become progressively smaller toward the poles, leading to excessive polar resolution and unequal cell areas.

This structure has important numerical consequences. The convergence of meridians introduces coordinate singularities at the poles, which can require special treatment in numerical schemes [16]. Alternative discretizations, such as reduced latitude–longitude grids and geodesic grids based on triangular or hexagonal cells, have therefore been introduced to reduce these issues or better suit particular numerical methods [17]. Despite these limitations, latitude–longitude grids remain widely used because of their simplicity, compatibility with existing datasets, and convenient tensor-product structure. However, their unequal cell areas mean that any aggregation over grid points, whether computing losses, metrics, or spatial averages, must account for the varying cell area to avoid a systematic polar bias.

One way to avoid the polar over-representation and unequal cell areas associated with latitude–longitude grids is to use an equal-area pixelization of the sphere. HEALPix (Hierarchical Equal Area iso-Latitude Pixelization) was introduced by Górski et al. as a spherical discretization combining three properties: equal-area pixels, iso-latitude sampling, and a hierarchical structure [18]. These properties are designed to support efficient numerical operations on spherical data, such as spherical harmonic transforms, convolutions, wavelet decompositions and nearest-neighbour searches.

At the base resolution, HEALPix partitions the sphere into twelve curvilinear quadrilateral pixels of equal area. These consist of four equatorial base pixels and eight polar base pixels, with four pixels residing in each polar cap. Higher-resolution partitions are generated recursively by subdividing each pixel into four equal-area child pixels. The resolution is specified by the parameter N_{side}, which denotes the number of subdivisions along the side of each base-resolution pixel. Figure 2.1 illustrates the resulting hierarchical refinement for several values of N_{side}.

For a given value of N_{side}, the total number of pixels is

$$N_{\text{pix}} = 12N_{\text{side}}^2.$$

Since the surface area of the unit sphere is 4π, the solid angle associated with each pixel is

$$\Omega_{\text{pix}} = \frac{4\pi}{N_{\text{pix}}} = \frac{\pi}{3N_{\text{side}}^2}.$$

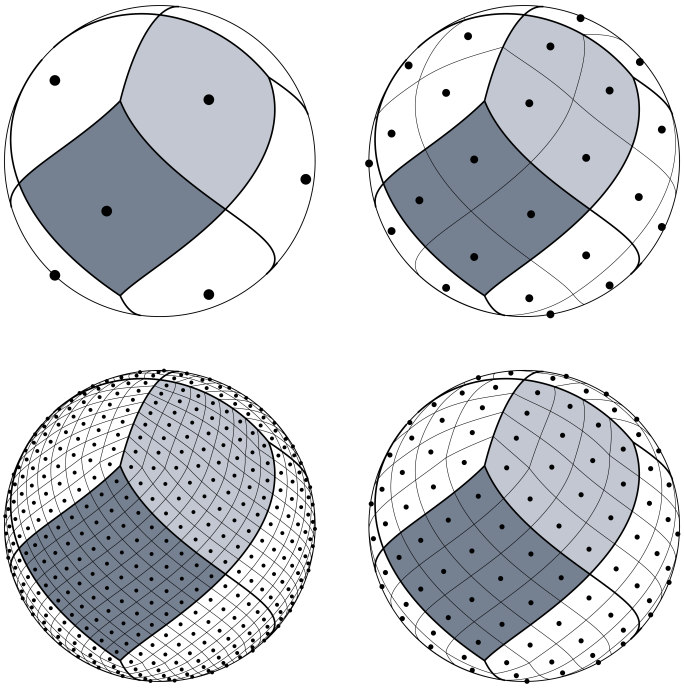

Figure 2.1: HEALPix pixelization of the sphere for different values of N_{side}. The panels show the grids for $N_{\text{side}} = 1$ in the top-left panel, $N_{\text{side}} = 2$ in the top-right panel, $N_{\text{side}} = 4$ in the bottom-right panel, and $N_{\text{side}} = 8$ in the bottom-left panel.

HEALPix also defines two common pixel-ordering schemes: RING and NESTED. In the RING scheme, pixels are ordered ring by ring from north to south, which is convenient for operations based on the iso-latitude structure, such as spherical harmonic analysis. In the NESTED scheme, the hierarchical subdivision of the base pixels is reflected directly in the pixel indexing. This makes the NESTED ordering useful for multiresolution operations, nearest-neighbour searches, and wavelet-based methods. The choice of ordering therefore does not change the geometry of the grid, but it affects how efficiently different operations can be implemented.

2.5 Transformer models and self-attention

Transformer models were introduced for natural language processing as an alternative to recurrent and convolutional sequence models [19]. Their central mechanism is self-attention, which updates each token by comparing it with other tokens in the sequence making it possible to capture long-range dependencies while allowing token-wise computations to be parallelized.

Since their introduction, Transformer-based architectures have been adapted far beyond language modeling, including computer vision, speech processing, time-series modeling, and scientific machine learning. This flexibility comes from representing data as sequences of tokens, such as subwords in language models, image patches in vision models, or spatial and temporal patches in physical modeling tasks.

The central concept of the transformer is the self-attention mechanism, which works by linearly projecting each embedded token into a query, a key, and a value representation. Let $X \in \mathbb{R}^{N \times d}$ be a sequence of N token embeddings each of dimension d. The embeddings are projected into queries, keys and value representations as follows:

$$Q = XW_Q, \qquad K = XW_K, \qquad V = XW_V,$$

where $W_Q, W_K \in \mathbb{R}^{d \times d_k}$, $W_V \in \mathbb{R}^{d \times d_v}$, are learned projection matrices. The attention output is then

$$Y = \text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V,$$

where the softmax is applied row-wise. The matrix $QK^\top \in \mathbb{R}^{N \times N}$ contains the dot products between every query and every key. Its (i, j) entry, $q_i^\top k_j$, can be understood as measuring the compatibility between token i, represented by its query, and token j, represented by its key. The scaling by $\sqrt{d_k}$ prevents these dot products from becoming too large as the key dimension increases, which stabilizes the softmax operation.

In some architectures, the attention pattern is restricted by adding a mask before the softmax:

$$\text{Attention}(Q, K, V; M) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + M\right) V.$$

Here $M_{ij} = 0$ for allowed interactions and $M_{ij} = -\infty$ for disallowed interactions. This form is useful both for causal attention, where future tokens are hidden, and for local or windowed attention, where tokens are only allowed to attend within a prescribed neighbourhood.

Rather than using a single attention operation, Transformers typically use multi-head attention (MHA). For each head $h = 1, \ldots, H$, separate query, key, and value projections are learned:

$$Q^{(h)} = XW_Q^{(h)}, \qquad K^{(h)} = XW_K^{(h)}, \qquad V^{(h)} = XW_V^{(h)}.$$

Each head computes its own attention output,

$$Y^{(h)} = \text{Attention}\left(Q^{(h)}, K^{(h)}, V^{(h)}\right).$$

The head outputs are then concatenated and mapped back to the model dimension by a learned output projection W_O:

$$\text{MHA}(X) = \text{Concat}\left(Y^{(1)}, \ldots, Y^{(H)}\right) W_O.$$

Using several heads allows the model to represent different types of token interactions in parallel, since each head has its own learned projections.

A Transformer encoder block combines multi-head self-attention with a position-wise multilayer perceptron (MLP), layer normalization (LN), and residual connections. With layer normalization applied before each sublayer, the block can be written as:

$$\tilde{X} = X + \mathrm{MHA}(\mathrm{LN}(X)),$$

$$X_{\mathrm{out}} = \tilde{X} + \mathrm{MLP}(\mathrm{LN}(\tilde{X})).$$

The attention layer mixes information between tokens, while the MLP acts independently on each token representation. The residual connections help preserve information across layers and improve optimization in deep networks.

2.6 Vision Transformer

The Vision Transformer adapts the Transformer encoder to images by treating image patches as tokens [20]. Given an image or gridded field

$$x \in \mathbb{R}^{H \times W \times C},$$

with height H, width W, and C channels, the input is divided into non-overlapping patches of size $P \times P$. This gives $N = \frac{HW}{P^2}$ patches, assuming H and W are divisible by P. Each patch is flattened into a vector $p_i \in \mathbb{R}^{P^2 C}$ and then mapped to the transformer embedding dimension d using a learned linear projection:

$$z_i = p_i E, \qquad E \in \mathbb{R}^{(P^2 C) \times d}.$$

The resulting patch embeddings form a token sequence

$$Z = (z_1, \ldots, z_N) \in \mathbb{R}^{N \times d}.$$

Self-attention is then applied to this sequence inside the transformer encoder using the attention mechanism described in the previous section, with Z used as the input token matrix in place of X. Positional information must be added in order for the model to distinguish where each patch came from. This is commonly done by adding learned or fixed positional embeddings to the patch embeddings before they are passed through the Transformer encoder.

A limitation of global self-attention is its quadratic cost in the number of tokens. For N patches, the attention matrix has size $N \times N$, giving complexity $\mathcal{O}(N^2)$. This becomes expensive for high-resolution spatial fields, where the number of patches is large.

2.7 Swin Transformer

The Swin Transformer addresses the computational cost of global attention by replacing global self-attention with local self-attention computed inside non-overlapping windows [21]. Instead of allowing every token to attend to every other token, the feature map is partitioned into windows of size $M \times M$, and self-attention is computed separately within each window. For N tokens, this changes the attention complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(NM^2)$, which is linear in N for fixed window size M. The trade-off is that each layer only mixes information within each local window.

To allow information to pass across window boundaries, Swin alternates between window-based multi-head self-attention (W-MSA) and shifted-window multi-head self-attention (SW-MSA). In W-MSA, attention is computed within fixed non-overlapping windows. In the following SW-MSA block, the window partition is shifted by approximately half the window size, so tokens that previously belonged to different windows may attend to each other.

A pair of Swin blocks can be written as

$$\hat{Z}^l = Z^{l-1} + \text{W-MSA}(\text{LN}(Z^{l-1})), \qquad Z^l = \hat{Z}^l + \text{MLP}(\text{LN}(\hat{Z}^l)),$$

$$\hat{Z}^{l+1} = Z^l + \text{SW-MSA}(\text{LN}(Z^l)), \qquad Z^{l+1} = \hat{Z}^{l+1} + \text{MLP}(\text{LN}(\hat{Z}^{l+1})).$$

The shifted windows are implemented efficiently by cyclically shifting the feature map before window partitioning. An attention mask is applied to prevent tokens from attending across the wrap-around boundaries introduced by the shift.

Swin further builds a hierarchical representation using patch merging layers. After a number of Swin blocks, neighbouring 2×2 groups of tokens are concatenated and linearly projected, reducing the spatial resolution from

$$H_s \times W_s \quad \text{to} \quad \frac{H_s}{2} \times \frac{W_s}{2},$$

while typically increasing the channel dimension from C_s to $2C_s$. This gives Swin a multi-scale structure similar to the feature hierarchies used in convolutional networks. Within each window, Swin also adds a learned relative position bias to the attention scores,

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}} + B\right) V,$$

where B depends on the relative displacement between token positions inside the same window. This allows the attention mechanism to encode local spatial relationships without relying only on absolute positional embeddings.

2.8 PEAR

PEAR, Pangu Equal ARea, is a volumetric Swin Transformer for medium-range weather forecasting on the sphere, trained and evaluated on ERA5 Lite data. It builds on Pangu-Weather, a volumetric Swin Transformer that operates on equiangular data, but instead works natively on the HEALPix grid. To do so, it adapts patch embedding, windowed attention, shifting, and hierarchical downsampling to the HEALPix representation. The result is a parameter-efficient spherical architecture that improves forecasting performance over equiangular-grid baselines such as Pangu-Weather [4], GraphCast [22], and FengWu [23] across most variables and lead times [5]. PEAR's architecture will be modified to handle the new task of climate model emulation.

The original PEAR model uses separate surface and upper-atmosphere inputs with shapes $x_{\text{surf}} \in \mathbb{R}^{12N_{\text{side}}^2 \times 4}$ and $x_{\text{upper}} \in \mathbb{R}^{12N_{\text{side}}^2 \times 13 \times 5}$. The two inputs are patch embedded separately using a 1D convolution with kernel size and stride 16 for the surface variables and a 2D convolution with kernel size and stride $(16, 2)$ for the upper variables. This gives embedded tensors of shape $\left(\frac{3}{4}N_{\text{side}}^2, 48\right)$ and $\left(\frac{3}{4}N_{\text{side}}^2, 7, 48\right)$, which are concatenated into a single latent tensor $\left(\frac{3}{4}N_{\text{side}}^2, 8, 48\right)$.

The latent tensor is processed by shifted window attention blocks. In the first resolution stage, PEAR applies two attention blocks with 6 attention heads and embedding dimension 48. Attention is computed inside local windows of size (W_{hp}, W_d), where W_{hp} is the window size along the HEALPix dimension and W_d is the window size along the vertical dimension. Each window therefore contains $W_{\text{hp}}W_d$ latent voxels. Every other block shifts the windows by approximately half the window size.

After the first attention stage, PEAR applies a single downsampling step along the HEALPix dimension, reducing the spatial resolution while increasing the embedding dimension. The tensor is changed from $\left(\frac{3}{4}N_{\text{side}}^2, 8, 48\right)$ to $\left(\frac{3}{16}N_{\text{side}}^2, 8, 96\right)$, followed by a bottleneck stage of 12 shifted-window attention blocks with 12 attention heads. The decoder mirrors the downsampling operation and the tensor is upsampled from $\left(\frac{3}{16}N_{\text{side}}^2, 8, 96\right)$ back to $\left(\frac{3}{4}N_{\text{side}}^2, 8, 48\right)$. Two additional shifted window attention blocks with 6 heads are then applied at the recovered resolution. The output of this final attention stage is concatenated with the output of the first high-resolution attention stage, giving a tensor of shape $\left(\frac{3}{4}N_{\text{side}}^2, 8, 96\right)$ before patch recovery.

Patch recovery maps the latent tensor back to separate surface and upper-variable outputs. The tensor $x \in \mathbb{R}^{\frac{3}{4}N_{\text{side}}^2 \times 8 \times 48}$ is split along the vertical dimension into one surface level and seven upper levels and decoded with a 1D transpose convolution for the surface, while the upper part is decoded with a 2D transpose convolution, giving outputs $\hat{x}_{\text{surf}} \in \mathbb{R}^{12N_{\text{side}}^2 \times 4}$ and $\hat{x}_{\text{upper}} \in \mathbb{R}^{12N_{\text{side}}^2 \times 13 \times 5}$. An overview of the PEAR architecture can be seen in Table 2.1 and Figure 2.2. How the PEAR architecture has been adapted for ClimateSet is described in Section 3.5.

Table 2.1: Original PEAR model architectural overview showing the depth, tensor shape and number of attention heads per layer.

Nr	Layer/Block	Depth	Tensor shape	Attention heads
	Input	1	$\begin{cases} (12n_{\text{side}}^2, 4) \\ (12n_{\text{side}}^2, 13, 5) \end{cases}$	
1	Patch embed	1	$(3/4 n_{\text{side}}^2, 8, 48)$	
2	Windowed attention	2	$(3/4 n_{\text{side}}^2, 8, 48)$	6
3	Downsample	1	$(3/16 n_{\text{side}}^2, 8, 96)$	
4	Windowed attention	12	$(3/16 n_{\text{side}}^2, 8, 96)$	12
5	Upsample	1	$(3/4 n_{\text{side}}^2, 8, 48)$	
6	Windowed attention	2	$(3/4 n_{\text{side}}^2, 8, 48)$	6
	Concatenate 6 & 2	1	$(3/4 n_{\text{side}}^2, 8, 96)$	
	Patch recovery	1	$\begin{cases} (12n_{\text{side}}^2, 4) \\ (12n_{\text{side}}^2, 13, 5) \end{cases}$	

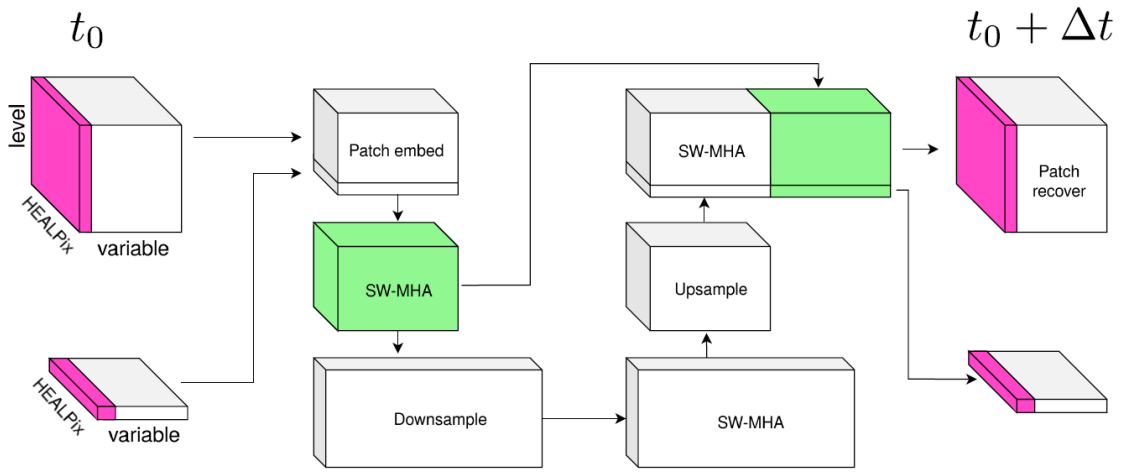

Figure 2.2: A schematic of the original PEAR architecture. Adapted with permission from [5]. The blocks on the left represent PEAR's input tensors, consisting of upper-air and surface variables. These are patch-embedded and passed through shifted-window multi-head attention (SW-MHA) blocks with skip connections, followed by downsampling, additional SW-MHA blocks, upsampling, and final SW-MHA blocks. A final patch-recovery stage then reconstructs the output tensors and produces a one-timestep-ahead prediction.

3

Methodology

3.1 ClimateSet

ClimateSet is a large-scale, machine-learning-ready dataset designed to support climate model emulation. It was introduced to address the lack of standardized, multi-model climate datasets for machine learning, where previous benchmarks often relied on outputs from only one or a small number of climate models. ClimateSet contains inputs and outputs from 36 CMIP6 climate models, sourced from the Input4MIPs and CMIP6 archives, and provides a consistent preprocessing pipeline for constructing emulation datasets across models and scenarios [6]. In this work, the publicly available ClimateSet subset is used, consisting of 21 climate models.

The input variables in ClimateSet are derived from Input4MIPs, the standardized CMIP input-data collection that provides prescribed forcing and boundary-condition datasets for CMIP experiments. These datasets include quantities such as greenhouse gas concentrations, aerosol and land-use-related forcings, and other external drivers that are prescribed to climate models rather than simulated internally. The ClimateSet input dataset consists of global monthly emission maps for four primary climate forcing agents: carbon dioxide (CO_2), methane (CH_4), sulfur dioxide (SO_2), and black carbon (BC), representing both long-lived greenhouse gases and shorter-lived aerosol-related forcings. The dataset includes one historical scenario and four future Shared Socioeconomic Pathways: SSP1-2.6, SSP2-4.5, SSP3-7.0, and SSP5-8.5. For each scenario, the inputs describe externally prescribed emissions, while the target variables represent the climate response simulated by the corresponding CMIP6 models. The target outputs in ClimateSet are global, monthly averaged maps of near-surface air temperature and precipitation from the CMIP6 ScenarioMIP archive. These two variables define the prediction task considered in this work.

All variables in the core ClimateSet collection are standardized to a global spatial resolution of approximately 250 km, corresponding to a 144×96 longitude–latitude grid with resolution $2.5^\circ \times 1.875^\circ$. The temporal coverage extends from 1850 to 2014 for historical simulations and from 2015 to 2100 for future projections, with all variables provided at monthly frequency.

ClimateSet includes several machine learning baselines evaluated on the ClimateSet emulation task. In this work, we use the single-emulator benchmark, in which the machine learning models are trained and evaluated separately for each climate model. The baseline models considered are a UNet model [24], a Convolutional LSTM [25], and a ClimaX [26] implementation. Of these, only the Convolutional LSTM explicitly models temporal dependencies across time steps, while the UNet and ClimaX baselines, as implemented in ClimateSet, treat each time step independently.

In this setup, the models are trained on the SSP1-2.6, SSP3-7.0, and SSP5-8.5 scenarios, with 10% of the training data held out for validation, and evaluated on the SSP2-4.5 scenario. The reported metric is the RMSE computed on normalized labels, averaged over near-surface air temperature and precipitation.

In the original ClimateSet baseline implementation, each baseline model is trained and evaluated independently on 15 of the 21 available climate models using three random seeds per climate model. In this work, we extend this evaluation to five random seeds per climate model to obtain more robust performance estimates.

3.2 Discretizing ClimateSet on HEALPix

To use PEAR with ClimateSet, the input and target fields must first be represented on a HEALPix grid. Given the much coarser spatial resolution of ClimateSet as compared to ERA5 Lite it is not necessary to grid at $N_{side} = 64$. The choice of N_{side} should therefore be matched to the native ClimateSet resolution as closely as possible.

To compare the HEALPix-gridded data fairly against the equiangular-grid baselines, we want to avoid giving the regridded data an artificial advantage through higher spatial resolution in parts of the globe. Since an equiangular longitude–latitude grid has its largest grid-cell area at the equator, an appropriate HEALPix resolution should have pixels with a solid angle close to, or slightly larger than, the equatorial grid cells. This approximately saturates the native angular resolution without introducing substantial oversampling. For a longitude–latitude grid, using colatitude θ and longitude ϕ, the solid angle of a cell is

$$\Omega_{\text{data}} = \int_{\phi_1}^{\phi_2} \int_{\theta_1}^{\theta_2} \sin\theta \, d\theta \, d\phi = \Delta\phi \left(\cos\theta_1 - \cos\theta_2\right),$$

where $\Delta\phi = \phi_2 - \phi_1$. The cell area is largest at the equator, where $\sin\theta$ is maximal. For the ClimateSet grid, with

$$\Delta\phi = 2.5^\circ, \qquad \Delta\theta = 1.875^\circ,$$

the maximum cell solid angle is obtained for a cell centered at the equator. The equatorial cell borders are therefore at

$$\theta_1 = 90^\circ - \frac{1.875^\circ}{2} \qquad \theta_2 = 90^\circ + \frac{1.875^\circ}{2}$$

with angles converted to radians, gives

$$\Omega_{\text{data,max}} = 1.4278 \times 10^{-3} \text{ sr}.$$

The HEALPix grid has equal-area pixels, with each pixel covering the solid angle

$$\Omega_{\text{hp}} = \frac{\pi}{3N_{\text{side}}^2}.$$

Matching the HEALPix pixel area to the largest native ClimateSet grid-cell area gives the condition

$$\Omega_{\text{hp}} \gtrsim \Omega_{\text{data,max}},$$

or equivalently

$$N_{\text{side}} \lesssim \sqrt{\frac{\pi}{3\Omega_{\text{data,max}}}}.$$

Substituting the ClimateSet value gives

$$N_{\text{side}} \lesssim 27.$$

Since valid HEALPix resolutions are powers of two, the closest practical choices are $N_{\text{side}} = 16$ and $N_{\text{side}} = 32$. According to the criterion above, $N_{\text{side}} = 32$ is somewhat finer than the native grid at the equator, leading to limited local oversampling. Using $N_{\text{side}} = 16$, however, would reduce the native ClimateSet grid from $144 \times 96 = 13824$ spatial points to $12 \cdot 16^2 = 3072$ HEALPix pixels, resulting in a substantial loss of spatial detail. In contrast, $N_{\text{side}} = 32$ gives $12 \cdot 32^2 = 12288$ HEALPix pixels, which is closer to and still slightly below the original number of spatial points. Although this resolution is locally finer than the native grid in some regions, this was judged to be an acceptable trade-off against the much stronger downsampling implied by $N_{\text{side}} = 16$. $N_{\text{side}} = 32$ is therefore used despite not fully satisfying the criterion above.

In practice, the ClimateSet fields are regridded in a custom PyTorch dataset class. For a given N_{side}, the $N_{\text{pix}} = 12N_{\text{side}}^2$ HEALPix pixel centres are generated using the `healpix` package in NESTED ordering. The native longitude–latitude fields are then interpolated to these pixel centres using `xarray.DataArray.interp`, producing flattened fields of shape (T, N_{pix}) before they are passed to the model.

As a check on the effect of this regridding step, the training-set statistics were compared before and after interpolation to HEALPix. The resulting differences were very small where for surface temperature, the mean and standard deviation changed by less than 0.002% and 0.2%, respectively, while for precipitation the mean changed by less than 0.03% and the standard deviation decreased by approximately 2.7%. This suggests that the regridding does not substantially alter the global distribution of the target fields. The slightly larger change for precipitation is likely due to its stronger spatial variability, which makes it more sensitive to interpolation.

3.3 Normalization and evaluation

To stabilize training, the target variables are normalized before being passed to the model. This is done for both the HEALPix and equiangular-grid data. Since the ClimateSet baselines report metrics on normalized target variables, and the same convention will be used for PEAR.

For a target variable y, the normalized label is defined as

$$\tilde{y} = \frac{y - \mu_y}{\sigma_y},$$

where μ_y and σ_y are computed over the training data. Since the target fields are defined on the sphere, these statistics should correspond to spherical averages rather than simple averages over grid indices.

For the equiangular longitude–latitude grid, the grid cells do not have equal area. A uniform average over grid points would overweight the polar regions, where grid cells represent smaller physical areas. The normalization statistics are therefore computed using latitude weights

$$w_i = \cos \varphi_i,$$

where φ_i is the latitude of row i. For

$$y \in \mathbb{R}^{T \times N_{\text{lat}} \times N_{\text{lon}}},$$

the weighted mean is

$$\mu_y = \frac{\sum_{t=1}^{T} \sum_{i=1}^{N_{\text{lat}}} \sum_{j=1}^{N_{\text{lon}}} w_i y_{t,i,j}}{T N_{\text{lon}} \sum_{i=1}^{N_{\text{lat}}} w_i},$$

and the corresponding weighted variance is

$$\sigma_y^2 = \frac{\sum_{t=1}^{T} \sum_{i=1}^{N_{\text{lat}}} \sum_{j=1}^{N_{\text{lon}}} w_i \left(y_{t,i,j} - \mu_y\right)^2}{T N_{\text{lon}} \sum_{i=1}^{N_{\text{lat}}} w_i}.$$

For the HEALPix representation, all pixels have equal area. Given

$$y \in \mathbb{R}^{T \times N_{\text{pix}}},$$

the mean and variance are therefore computed uniformly over time and pixels,

$$\mu_y = \frac{1}{T N_{\text{pix}}} \sum_{t=1}^{T} \sum_{p=1}^{N_{\text{pix}}} y_{t,p},$$

$$\sigma_y^2 = \frac{1}{T N_{\text{pix}}} \sum_{t=1}^{T} \sum_{p=1}^{N_{\text{pix}}} \left(y_{t,p} - \mu_y\right)^2.$$

The loss and evaluation metrics are computed using the same spherical averaging principle. For HEALPix, the mean squared error is

$$\mathrm{MSE}_{\mathrm{hp}} = \frac{1}{TN_{\mathrm{pix}}} \sum_{t=1}^{T} \sum_{p=1}^{N_{\mathrm{pix}}} \left(\hat{\tilde{y}}_{t,p} - \tilde{y}_{t,p} \right)^2,$$

with

$$\mathrm{RMSE}_{\mathrm{hp}} = \sqrt{\mathrm{MSE}_{\mathrm{hp}}}.$$

For the equiangular grid, the latitude-weighted mean squared error is

$$\mathrm{MSE}_{\mathrm{eq}} = \frac{\sum_{t=1}^{T} \sum_{i=1}^{N_{\mathrm{lat}}} \sum_{j=1}^{N_{\mathrm{lon}}} w_i \left(\hat{\tilde{y}}_{t,i,j} - \tilde{y}_{t,i,j} \right)^2}{TN_{\mathrm{lon}} \sum_{i=1}^{N_{\mathrm{lat}}} w_i},$$

and

$$\mathrm{RMSE}_{\mathrm{eq}} = \sqrt{\mathrm{MSE}_{\mathrm{eq}}}.$$

This gives comparable metrics for the HEALPix and equiangular representations, since both approximate averages over the sphere rather than averages over the chosen grid indices.

3.4 Baseline re-implementation

ClimateSet provides benchmark code and results for U-Net, ConvLSTM, and ClimaX. However, documented errors in the public evaluation pipeline affected the variable splitting and latitude weighting used for the baseline metrics. As a result, the reported baseline metrics do not correspond to variable-wise, area-weighted global RMSE. These issues have been acknowledged by the ClimateSet authors and noted in subsequent work [27, 3]. The original baseline numbers were therefore not used directly; instead, the baseline models were reimplemented using a corrected data loading, preprocessing, training, and evaluation pipeline.

The main issue was the splitting of predictions and targets by variable. ClimateSet tensors have shape

$$[B, T, C, H, W],$$

where C is the variable dimension. The original implementation treated C as if it were the final dimension, causing the split to occur along a spatial axis. The resulting evaluation tensors had shape

$$[B, T, C, H, 1]$$

rather than

$$[B, T, 1, H, W].$$

Thus, the reported per-variable RMSE was computed over a single spatial band still containing both target variables, rather than over the full global field for each variable.

A second issue concerned the latitude-weighted RMSE for the equiangular grid. The original implementation applied the latitude weights along the longitude dimension, so the metric did not represent an area-weighted global error. In the corrected implementation, tensors are split along the channel dimension and latitude weights are applied along the latitude dimension. The preprocessing was also adjusted to use latitude-area-weighted normalization statistics for the equiangular grid, as described above. The three baselines were reimplemented using a new data loader and the same training and evaluation backbone as the PEAR experiments, the parameters were kept as in the ClimateSet implementation, with the exception of the scheduler for the ConvLSTM which when removed improved its performance. ClimaX was used without pretraining, as [6] found it did not improve performance. The number of trainable parameter for the baseline models, alongside the final PEAR model, can be seen in Table 3.1.

Model	Trainable parameters
ConvLSTM	730K
PEAR	4.3M
UNet	18.3M
ClimaX	106.6M

Table 3.1: Number of trainable parameters for each of the implemented machine learning models.

3.5 Adapting PEAR to climate model emulation

To adapt PEAR to ClimateSet, the architecture is modified at the input and output stages to match the structure of the climate emulation task. The original PEAR model is designed with separate inputs for surface variables and upper-atmosphere variables defined over multiple vertical levels and predicts the same set of variables at a future timestep as the output. In ClimateSet, the input instead consists of four climate forcing agents and the output of two target climate variables. The same structure with shifted-window attention, downsampling, bottleneck, upsampling, and skip connection is otherwise retained, but all windows have only one element in the depth direction.

These differences are addressed at the input stage by removing the upper-atmosphere branch and reducing the patch embedding to a single surface embedding applied to the ClimateSet input tensor $x \in \mathbb{R}^{12N_{\text{side}}^2 \times C_{\text{in}}}$, where C_{in} is the number of input forcing channels. A 1D convolution along the HEALPix dimension, with kernel size and stride 16, maps the input to a latent tensor of shape $\left(\frac{3}{4}N_{\text{side}}^2, d_{\text{emb}}\right)$, where d_{emb} is the embedding dimension.

Since the ClimateSet input has no vertical structure, the volumetric dimension used in the original PEAR architecture is replaced by a singleton depth dimension. The latent tensor is therefore treated as $x \in \mathbb{R}^{\frac{3}{4}N_{\text{side}}^2 \times 1 \times d_{\text{emb}}}$, corresponding to a vertical attention depth $D = 1$ throughout the transformer blocks.

After the final attention stage, patch recovery is performed using a single 1D transpose convolution. This maps the latent representation back to the ClimateSet target variables, $\hat{x} \in \mathbb{R}^{12N_{\text{side}}^2 \times C_{\text{out}}}$, where C_{out} is the number of predicted climate channels.

Different embedding dimensions and backbone depths were considered during experimentation. The final configuration uses $d_{\text{emb}} = 48$, as in the original PEAR implementation, but doubles the transformer stage depths from $(2, 6, 6, 2)$ to $(4, 12, 12, 4)$. The resulting model has approximately 4.3M trainable parameters, making it larger than the original PEAR model but still comparatively small relative to the baselines considered in this work.

The adapted architecture can be seen in Figure 3.3, and the tensor shapes are summarized in Table 3.2. To get the best performance for PEAR significant ablations and hyperparameter tunings has been done. The primary variables considered were: learning rate, embedding dimensions, dropout and depth. As well as experimentation with variable weighing and separate output heads to see if improvements could be achieved in one variable by weighing its result higher in the loss function. To get a metric of which model configuration performed the best, an average RMSE has been calculated across all 15 climate models. For each climate model the value reported is the best epoch on the test set per seed, averaged over seeds. This value was then averaged across all climate models.

3.6 Multi-timestep models

In addition to the main PEAR adaptation, a causal temporal extension was implemented to investigate whether the PEAR depth dimension could be repurposed as a time dimension. In this model, the input is reformulated as a sequence of T consecutive climate states,

$$x \in \mathbb{R}^{T \times 12N_{\text{side}}^2 \times C_{\text{in}}}.$$

The temporal dimension is then used as the depth dimension in the PEAR backbone.

To embed the sequence input, the 1D convolution used in the single-timestep model is replaced by a 2D convolution over the temporal and HEALPix dimensions. The convolution uses kernel size and stride $(1, p)$, where $p = 16$ is the spatial patch size. This applies the same spatial patching as in the base model, while keeping each timestep separate in the temporal direction. The resulting latent tensor has shape

$$\left(T, \frac{3}{4}N_{\text{side}}^2, d_{\text{emb}}\right),$$

so the depth dimension entering the transformer blocks becomes $D = T$. The recovery layer is changed analogously: the 1D transpose convolution is replaced by a 2D transpose convolution with temporal kernel size and stride 1, preserving the temporal resolution and producing

$$\hat{x} \in \mathbb{R}^{T \times 12N_{\text{side}}^2 \times C_{\text{out}}}.$$

Table 3.2: Adapted PEAR model architectural overview showing the depth, tensor shape, and number of attention heads per layer.

Nr	Layer/Block	Depth	Tensor shape	Attention heads
	Input	1	$(12N_{\text{side}}^2, 4)$	
1	Patch embed	1	$(3/4N_{\text{side}}^2, 1, 48)$	
2	Windowed attention	4	$(3/4N_{\text{side}}^2, 1, 48)$	6
3	Downsample	1	$(3/16N_{\text{side}}^2, 1, 96)$	
4	Windowed attention	24	$(3/16N_{\text{side}}^2, 1, 96)$	12
5	Upsample	1	$(3/4N_{\text{side}}^2, 1, 48)$	
6	Windowed attention	4	$(3/4N_{\text{side}}^2, 1, 48)$	6
	Concatenate 6 & 2	1	$(3/4N_{\text{side}}^2, 1, 96)$	
	Patch recovery	1	$(12N_{\text{side}}^2, 2)$	

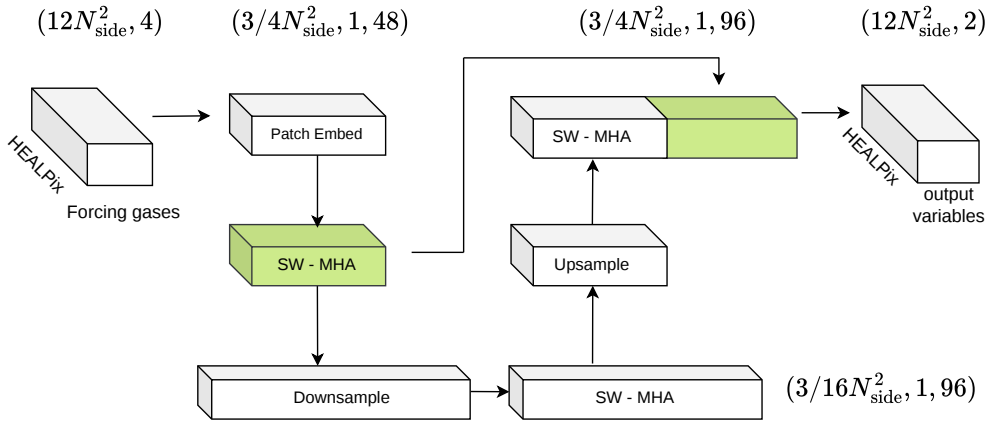

Table 3.3: Schematic of the adapted PEAR architecture with tensor dimensions for each layer. The input is changed to handle the climate forcing gases without a vertical dimension. The patch embedding and SW-MHA layers use a singleton vertical dimension throughout the architecture, and the output is adapted to return the two climate variables.

To prevent information flow from future timesteps, causal masking is added in the temporal direction inside the transformer blocks. Each attention window spans both the temporal depth D and the HEALPix dimension, containing

$$W = W_d \cdot W_{\text{hp}}$$

tokens in total, where W_d and W_{hp} are the window extents in the temporal and spatial dimensions respectively. Since tokens within a window can belong to different timesteps, causality must be defined at the level of individual tokens rather than for the window as a whole.

For each window w, every token is assigned a timestep index $\tau^{(w)}$. The causal mask is then defined by comparing the timestep indices of every query–key pair inside the window:

$$m_{ij}^{(w)} = \begin{cases} -\infty, & \tau_j^{(w)} > \tau_i^{(w)}, \\ 0, & \text{otherwise.} \end{cases}$$

This prevents token i from attending to token j whenever token j represents a later timestep. Tokens at the same timestep, as well as tokens from previous timesteps, remain visible.

The causal mask is added to the standard shifted-window attention mask before the softmax. The resulting attention operation is therefore restricted both by the local window structure of the Swin architecture and by the temporal ordering of the input sequence. After the softmax, all attention weights corresponding to future timesteps are suppressed to zero.

4

Results

4.1 ClimateSet updated baseline results

Table 4.1 reports updated baseline results for experiments following the ClimateSet single emulator benchmark setup. The table is organized in the same way as the baseline comparison in the original ClimateSet paper, but uses the updated evaluation pipeline with corrections to the tensor splitting and RMSE computation, where the split was performed along the intended variable dimension, and RMSE on the longitude–latitude grid was computed with latitude weighting.

The reported values are averaged over five independent runs, rather than the three runs used in the original benchmark. The models were trained using the same general hyperparameter settings as in the ClimateSet implementation, except for ConvLSTM. For ConvLSTM, the learning-rate scheduler used in the original implementation was removed, since this gave better results in the updated experiments.

4.2 PEAR performance

Figure 4.1 compares the non-temporal ClimateSet baselines with the best-performing PEAR configuration across the 15 climate models and 5 random seeds. The figure reports RMSE both separately for surface temperature and precipitation, and as an aggregated score across the two variables.

For the aggregated RMSE, ClimaX obtains the lowest error for 10 of the 15 climate models, while UNet performs best for 3 models and PEAR for 2. However, PEAR outperforms UNet on 13 of the 15 climate models despite using roughly one quarter as many trainable parameters. Separating the results by variable shows a clearer difference between temperature and precipitation. For surface temperature, ClimaX achieves the lowest RMSE for 12 of the 15 climate models, with UNet performing best on the remaining 3. For precipitation, PEAR achieves the lowest RMSE for 8 of the 15 climate models, compared with 4 for ClimaX and 3 for UNet.

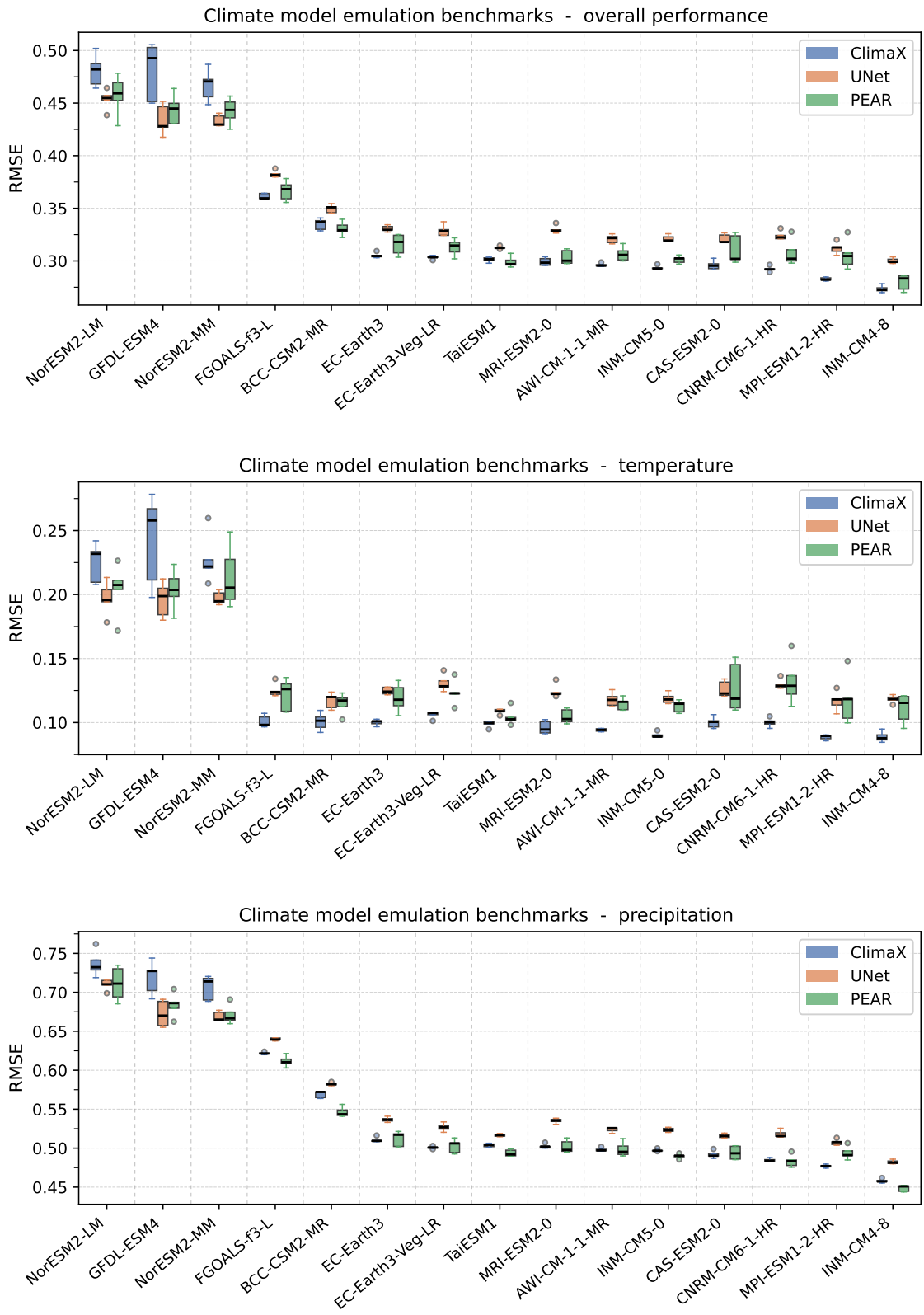

Figure 4.1: Box plots showing RMSE on normalized outputs for ClimaX, UNet, and PEAR across 15 ClimateSet climate models for 5 seeds each. The first plot shows overall performance, the second plot temperature performance and the third plot precipitation performance.

Table 4.1: Updated single-emulator baseline results following the ClimateSet benchmark setup. RMSE values are reported separately for near-surface air temperature (TAS), precipitation (PR), and their average for each climate model. Results are averaged over five independent runs and computed using the corrected evaluation pipeline, including variable-wise tensor splitting and latitude-weighted RMSE on the longitude–latitude grid.

	U-Net			ConvLSTM			ClimaX		
	TAS	PR	Avg	TAS	PR	Avg	TAS	PR	Avg
AWI-CM-1-1-MR	0.117	0.524	0.321	0.098	0.501	0.299	0.094	0.498	0.296
BCC-CSM2-MR	0.117	0.582	0.349	0.101	0.568	0.334	0.101	0.569	0.335
CAS-ESM2-0	0.126	0.515	0.321	0.104	0.495	0.299	0.098	0.490	0.294
CNRM-CM6-1-HR	0.130	0.518	0.324	0.110	0.488	0.299	0.100	0.485	0.292
EC-Earth3	0.125	0.537	0.331	0.107	0.518	0.312	0.100	0.510	0.305
EC-Earth3-Veg-LR	0.131	0.527	0.329	0.109	0.506	0.307	0.104	0.500	0.302
FGOALS-f3-L	0.125	0.640	0.383	0.102	0.622	0.362	0.101	0.621	0.361
GFDL-ESM4	0.196	0.672	0.434	0.090	0.552	0.321	0.242	0.719	0.480
INM-CM4-8	0.118	0.482	0.300	0.090	0.464	0.277	0.089	0.458	0.274
INM-CM5-0	0.119	0.523	0.321	0.093	0.504	0.298	0.090	0.497	0.293
MPI-ESM1-2-HR	0.117	0.508	0.312	0.091	0.486	0.288	0.089	0.477	0.283
MRI-ESM2-0	0.124	0.535	0.330	0.125	0.530	0.328	0.096	0.502	0.299
NorESM2-LM	0.197	0.710	0.454	0.093	0.589	0.341	0.225	0.737	0.481
NorESM2-MM	0.197	0.669	0.436	0.093	0.563	0.328	0.228	0.706	0.467
TaiESM1	0.109	0.517	0.313	0.110	0.511	0.310	0.099	0.504	0.301

4.3 Multi-timestep experiments

Figure 4.2 shows the per-variable and overall RMSE for ClimaX, ConvLSTM, the PEAR base model, and the PEAR multi-timestep model across 15 climate models and 5 random seeds. Compared with Figure 4.1, UNet is omitted so that the plot focuses on the models most relevant for evaluating temporal context. The PEAR multi-timestep model and ConvLSTM are included as the two multi-timestep models in the benchmark, while ClimaX is retained as the strongest single-timestep baseline. The PEAR base model is also included as a direct reference for assessing how the multi-timestep extension affects PEAR's performance.

In terms of overall RMSE, ClimaX still achieves the lowest error on 11 of the 15 climate models, followed by ConvLSTM performing best on 3 models and the PEAR multi-timestep model on 1. For surface temperature, ClimaX performs best on 12 of the 15 climate models, with ConvLSTM leading on the remaining 3. For precipitation, the PEAR models perform best overall, with the multi-timestep PEAR model achieving the lowest RMSE on 9 of the 15 climate models, while the PEAR base model and ConvLSTM each perform best on 3 models.

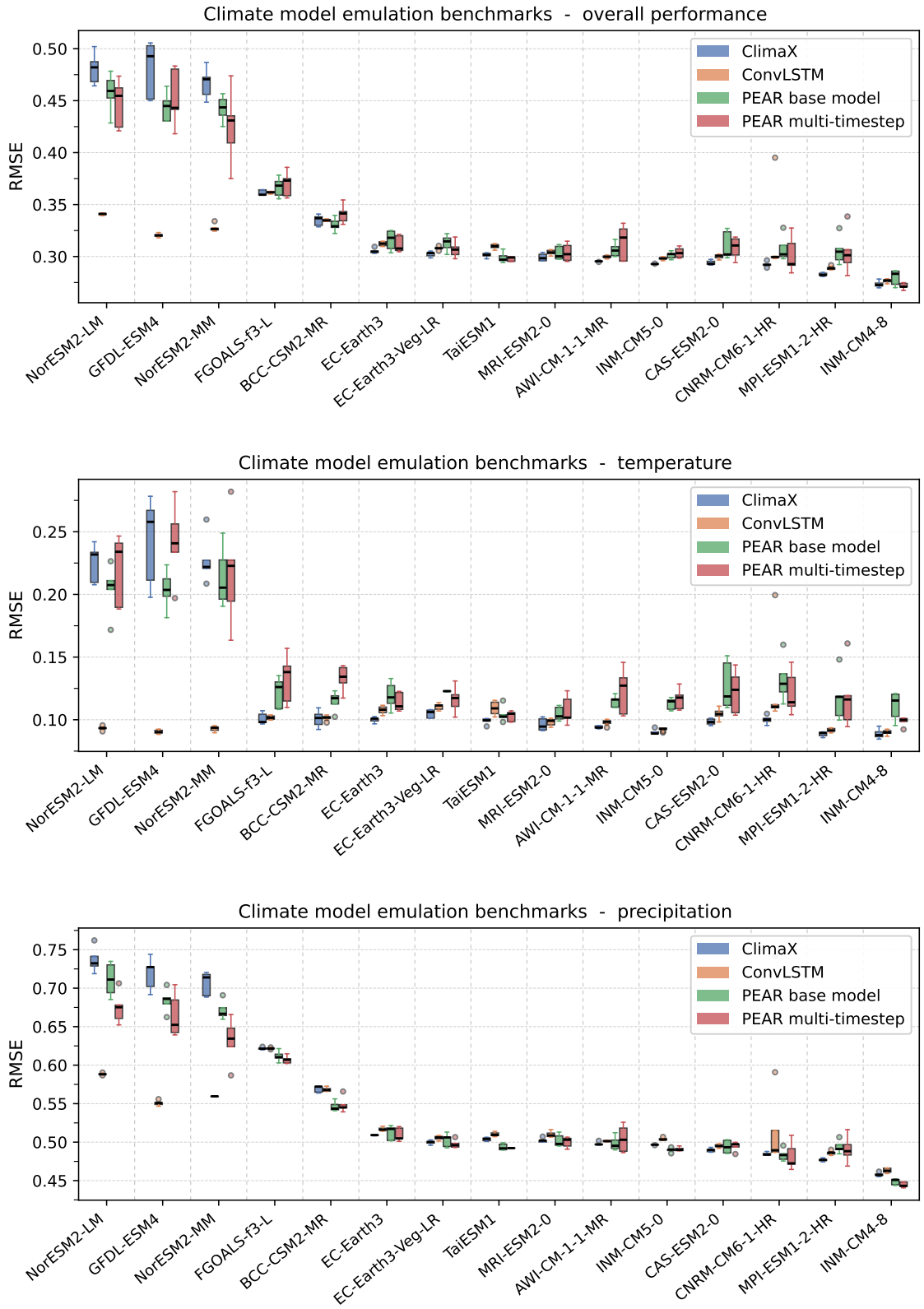

Figure 4.2: Box plots showing RMSE on normalized outputs for ClimaX, ConvLSTM and PEAR with and without multiple timesteps across 15 ClimateSet climate models for 5 seeds each. The first plot shows overall performance, the second plot temperature performance and the third plot precipitation performance.

4.4 Ablations

Table 4.2 reports the ablation results for the main PEAR climate-emulation configuration. The table includes experiments varying the learning rate, embedding dimension, dropout, depth, and attention-window size. The best-performing configuration from this ablation study is used as the PEAR model in the main comparison against the ClimateSet baselines.

Table 4.3 reports the ablation results for the multi-timestep PEAR configuration varying learning rate, attention-window size and batch size for the multi-timestep setting. All configurations are ranked by overall RMSE, computed as the average across the 15 climate models and 5 seeds per model.

C	Depth	Window	LR	Drop	RMSE	RMSE_{tas}	RMSE_{pr}
48	4/12	1×64	2e-4	0.0	0.3277	0.1215	0.5338
192	2/6	1×64	2e-4	0.0	0.3308	0.1264	0.5352
192	2/6	1×64	2.5e-4	0.0	0.3312	0.1282	0.5341
252	2/6	1×64	2e-4	0.0	0.3328	0.1292	0.5363
252	2/6	1×64	2e-4	0.1	0.3336	0.1332	0.5340
192	2/6	1×64	2.5e-4	0.1	0.3359	0.1355	0.5364
192	4/12	1×64	2e-4	0.00	0.3399	0.1396	0.5402
48	4/12	1×16	2e-4	0.00	0.3416	0.1409	0.5424
96	4/12	1×64	2e-4	0.00	0.3469	0.1460	0.5478
192	4/12	1×64	2.5e-4	0.00	0.3478	0.1523	0.5434
384	2/6	1×64	2e-4	0.00	0.3491	0.1498	0.5484
192	2/6	1×64	1e-4	0.00	0.3512	0.1469	0.5554
384	2/6	1×64	2e-4	0.10	0.3537	0.1559	0.5515
24	4/12	1×64	2e-4	0.00	0.3554	0.1548	0.5561
252	2/6	1×64	3e-4	0.00	0.3569	0.1625	0.5513
192	2/6	1×64	5e-5	0.00	0.3578	0.1542	0.5615
48	2/6	1×64	2e-4	0.00	0.3584	0.1572	0.5595
192	4/12	1×64	2.5e-4	0.10	0.3600	0.1676	0.5523
96	2/6	1×64	2e-4	0.00	0.3624	0.1608	0.5640
48	2/6	1×64	1e-4	0.00	0.3639	0.1620	0.5658
48	2/6	1×64	1.5e-4	0.00	0.3647	0.1618	0.5675
24	2/6	1×64	2e-4	0.00	0.3787	0.1821	0.5753
48	4/12	1×64	1e-4	0.00	0.3867	0.1913	0.5822
192	2/6	1×64	2.5e-4	0.05	0.3872	0.1862	0.5883
48	2/6	1×64	5e-4	0.00	0.3947	0.2015	0.5879
192	2/6	1×64	3e-4	0.00	0.3991	0.2094	0.5887
192	4/12	1×64	1e-4	0.00	0.4134	0.2235	0.6033

Table 4.2: Ablation over the standard PEAR climate-emulation configuration. Embedding dimensions following the symmetric pattern $[C, 2C, 2C, C]$; depth is reported as the (outer/inner) block count $[d, e, e, d]$, and window as (time×space). LR denotes learning rate and Drop denotes the dropout parameter. Metrics are RMSE averaged across 15 climate models and 5 seeds. Rows sorted by overall RMSE.

Window	LR	BS	RMSE	RMSE_{tas}	RMSE_{pr}
4×64	2e-4	6	0.3211	0.1192	0.5229
4×64	2e-4	12	0.3219	0.1221	0.5217
2×64	2e-4	12	0.3563	0.1607	0.5519
2×64	5e-4	12	0.3622	0.1767	0.5478
2×64	1e-4	12	0.4043	0.2070	0.6015

Table 4.3: Ablation over the multi-timestep PEAR variants. All runs use embedding dimensions $[48, 96, 96, 48]$ and no dropout; window is reported as (time×space), LR denotes learning rate and BS denotes batch size. Metrics are RMSE averaged across 15 climate models and 5 seeds. Rows sorted by overall RMSE.

5

Conclusions and future work

The main results, shown in Figure 4.1, indicate that the PEAR architecture is well suited for the task of climate model emulation. PEAR generally outperforms the UNet baseline for most climate models across both target variables, despite being roughly four times smaller. Although PEAR is mostly outperformed by ClimaX, this is not unexpected given ClimaX's considerably larger parameter count. A key result is that PEAR achieves the lowest precipitation RMSE for a majority of the climate models, outperforming ClimaX for 8 models and performing at a comparable level for most of the remaining models.

This strong precipitation performance makes the weaker results for surface temperature more notable. In contrast to precipitation, PEAR does not achieve the same relative performance on temperature. This is somewhat unexpected, since temperature is generally a smoother and more spatially coherent variable. Previous work has also shown that relatively simple approaches, such as linear pattern scaling, can perform well for temperature and in some cases outperform larger deep learning models [3]. One possible explanation is that PEAR's current architecture may be limited in its ability to capture sufficiently wide or global spatial dependencies. This is also suggested by the feature maps shown in Appendix A, where clear window-like structures remain visible before the final de-embedding stage. These structures indicate that information may not be propagating effectively between Swin windows, which could restrict the model's ability to learn global features.

The windowing scheme may therefore be an important direction for further architectural tuning. In the main experiments, the window size was kept at $[1, 64]$ with a shift of 4, following the original PEAR architecture. However, since the ClimateSet HEALPix data is gridded using $N_{\text{side}} = 32$ instead of $N_{\text{side}} = 64$ as in the original implementation, a smaller window size and shifting could more closely match the relative windowing structure of the original setup. Initial experiments with this change did not see any significant performance increases as compared to the other PEAR configurations. Nevertheless, the visible window structure in the feature maps suggests that the windowing scheme may still affect how well information is exchanged across the sphere, and that further architectural experimentation is needed to evaluate this properly.

The ablation experiments provide further context for this interpretation. Increasing the size of PEAR does not immediately lead to better performance, suggesting that the observed limitations are not simply caused by insufficient model capacity. One possible explanation is that the smaller model already captures most of the useful structure available to the current architecture, so that additional parameters do not provide a clear benefit. Although the larger configurations were trained for substantially longer, this did not lead to clear improvements, suggesting that the lack of performance gain is not due to insufficient training time. Different hyperparameter settings may still be needed for the larger models, but within the configurations tested here, increased capacity did not translate into better performance. This makes it less likely that the weaker temperature performance would be explained by model size alone. If limited capacity were the main issue, some improvement would be expected when increasing the size of the model. This instead supports the hypothesis that the limitation is more likely related to architectural constraints, such as how information is propagated across windows.

5.1 Future work

There is still substantial work that can be done to further develop PEAR for climate model emulation. A first direction is to investigate the weaker temperature performance. The results suggest that this may be related to how information is propagated across the windows, rather than model capacity. Further work should therefore consider architectural changes that improve information exchange between windows to help the model capture larger-scale spatial structure more effectively.

A second direction concerns the multi-timestep extension of PEAR, which has so far only seen initial experimentation. The temporal-attention PEAR model achieves a better ablation score than the single-step PEAR model, with an RMSE of 0.3211 compared with 0.3277, as shown in Table 4.2 and Table 4.3. The performance gains appear to be concentrated in only a few climate models, such as NorESM2-LM, INM-CM4-8, and NorESM2-MM, rather than showing consistent improvements across all models. These results indicate that incorporating multiple timesteps can improve PEAR's performance over the single-step formulation. However, even for the climate models where improvements are observed, the multi-timestep PEAR model still underperforms relative to the smaller ConvLSTM model, which is the only other multi-timestep model included in the benchmark. This suggests that there is still room for improvement and that further experimentation with temporal context is needed, either by refining the temporal-attention setup or by considering longer temporal windows.

Another possible direction would be to combine PEAR with a recurrent temporal module. In such a setup, PEAR could be applied to each timestep to extract spatial representations, while an additional recurrent component would model how these representations evolve over the input sequence. This would keep the main PEAR architecture focused on the spatial structure of the spherical fields, while adding a separate mechanism for temporal modeling.

A separate limitation of the current ClimateSet setup is that it only contains one ensemble member per climate model. As noted by [3], this can make it difficult to separate the forced climate response from internal variability. ClimateSet has stated the intention of extending the dataset to include multiple ensemble members per climate model [6], however this was not available at the time. There are other dataset that includes multiple ensemble members such as the Em-MPI that uses the MPI-ESM1.2-LR large ensemble, containing 50 realizations for each emission scenario [3]. Together with the multi-model ClimateSet experiments, such an evaluation would give a broader picture of PEAR's performance across multiple realizations of the same model. Due to time constraints, this experiment was not included in the present work and is left as a future direction.

ClimateSet also includes a super-emulator benchmarking task, which remains an interesting direction for future work. In this setting, a single machine learning model is trained on data from multiple climate models and conditioned to produce outputs for a specific target climate model, rather than training a separate emulator for each model. For PEAR, this could be explored by using a shared backbone to process the forcing inputs together with a multi-head decoder, where each climate model is assigned a lightweight decoding head and each sample is routed through its corresponding head. This would make it possible to evaluate whether PEAR can learn a more general forcing–response relationship across climate models, while potentially reducing the computational cost of training separate emulators.

Taken together, the directions outlined above point to many opportunities for further developing PEAR. While the current work establishes a strong baseline, many aspects of the architecture, temporal extension, and evaluation setup remain open for improvement. Each of these directions represents a promising avenue for continued development, and further tasks within the climate model emulation setting remain to be explored.

Bibliography

[1] D. Watson-Parris, "Machine learning for weather and climate are worlds apart," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 379, no. 2194, p. 20200098, 2021.

[2] P. Van Katwyk, B. Fox-Kemper, H. T. Hewitt, and K. J. Bergen, "Rewiring climate modeling with machine learning emulators," *Communications Earth & Environment*, vol. 7, no. 1, p. 107, 2026.

[3] B. Lütjens, R. Ferrari, D. Watson-Parris, and N. E. Selin, "The impact of internal variability on benchmarking deep learning climate emulators," *Journal of Advances in Modeling Earth Systems*, vol. 17, no. 8, p. e2024MS004619, 2025.

[4] K. Bi, L. Xie, H. Zhang, X. Chen, X. Gu, and Q. Tian, "Pangu-Weather: A 3d high-resolution model for fast and accurate global weather forecast," *arXiv preprint arXiv:2211.02556*, 2022.

[5] H. Linander, T. Tykesson, P. Rosso, C. Petersson, D. Persson, and J. E. Gerken, "PEAR: Equal area weather forecasting on the sphere," *arXiv preprint arXiv:2505.17720v3*, 2025.

[6] J. Kaltenborn, C. Lange, V. Ramesh, P. Brouillard, Y. Gurwicz, C. Nagda, J. Runge, P. Nowack, and D. Rolnick, "ClimateSet: A large-scale climate model dataset for machine learning," *Advances in Neural Information Processing Systems*, vol. 36, pp. 21 757–21 792, 2023.

[7] H.-O. Pörtner, D. C. Roberts, M. Tignor, E. S. Poloczanska, K. Mintenbeck, A. Alegría, M. Craig, S. Langsdorf, S. Löschke, V. Möller, A. Okem, and B. Rama, Eds., *Climate Change 2022: Impacts, Adaptation, and Vulnerability. Contribution of Working Group II to the Sixth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge, UK and New York, NY, USA: Cambridge University Press, 2022.

[8] A. Gettelman and R. B. Rood, *Demystifying climate models: A users guide to earth system models*. Springer, 2016.

[9] J. D. Neelin, *Climate change and climate modeling*. Cambridge University Press, 2010.

[10] G. A. Meehl, G. J. Boer, C. Covey, M. Latif, and R. J. Stouffer, "The coupled

model intercomparison project (CMIP)," *Bulletin of the American Meteorological Society*, vol. 81, no. 2, pp. 313–318, 2000.

[11] V. Eyring, S. Bony, G. A. Meehl, C. A. Senior, B. Stevens, R. J. Stouffer, and K. E. Taylor, "Overview of the coupled model intercomparison project phase 6 (CMIP6) experimental design and organization," *Geoscientific Model Development*, vol. 9, no. 5, pp. 1937–1958, 2016.

[12] B. C. O'Neill, C. Tebaldi, D. P. Van Vuuren, V. Eyring, P. Friedlingstein, G. Hurtt, R. Knutti, E. Kriegler, J.-F. Lamarque, J. Lowe *et al.*, "The scenario model intercomparison project (ScenarioMIP) for CMIP6," *Geoscientific Model Development*, vol. 9, no. 9, pp. 3461–3482, 2016.

[13] D. Watson-Parris, Y. Rao, D. Olivié, Ø. Seland, P. Nowack, G. Camps-Valls, P. Stier, S. Bouabid, M. Dewey, E. Fons *et al.*, "ClimateBench v1. 0: A benchmark for data-driven climate projections," *Journal of Advances in Modeling Earth Systems*, vol. 14, no. 10, p. e2021MS002954, 2022.

[14] T. Nguyen, J. Jewik, H. Bansal, P. Sharma, and A. Grover, "ClimateLearn: Benchmarking machine learning for weather and climate modeling," *Advances in Neural Information Processing Systems*, vol. 36, pp. 75 009–75 025, 2023.

[15] S. R. Cachay, V. Ramesh, J. N. Cole, H. Barker, and D. Rolnick, "ClimART: A benchmark dataset for emulating atmospheric radiative transfer in weather and climate models," *arXiv preprint arXiv:2111.14671*, 2021.

[16] A. Adcroft, J.-M. Campin, C. Hill, and J. Marshall, "Implementation of an atmosphere–ocean general circulation model on the expanded spherical cube," *Monthly Weather Review*, vol. 132, no. 12, pp. 2845–2863, 2004.

[17] T. T. Warner, *Numerical weather and climate prediction*. Cambridge University Press, 2010.

[18] K. M. Gorski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke, and M. Bartelmann, "HEALPix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere," *The Astrophysical Journal*, vol. 622, no. 2, pp. 759–771, 2005.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[21] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[22] R. Lam, A. Sanchez-Gonzalez, M. Willson, P. Wirnsberger, M. Fortunato, F. Alet, S. Ravuri, T. Ewalds, Z. Eaton-Rosen, W. Hu *et al.*, "Learning skillful medium-range global weather forecasting," *Science*, vol. 382, no. 6677, pp. 1416–1421, 2023.

[23] K. Chen, T. Han, J. Gong, L. Bai, F. Ling, J.-J. Luo, X. Chen, L. Ma, T. Zhang, R. Su *et al.*, "FengWu: Pushing the skillful global medium-range weather forecast beyond 10 days lead," *arXiv preprint arXiv:2304.02948*, 2023.

[24] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[25] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," *Advances in neural information processing systems*, vol. 28, 2015.

[26] T. Nguyen, J. Brandstetter, A. Kapoor, J. K. Gupta, and A. Grover, "ClimaX: A foundation model for weather and climate," *arXiv preprint arXiv:2301.10343*, 2023.

[27] J. Kaltenborn, J. Irvin, H. Saleem, Y. Hasson, and B. Lütjens, "Response to GitHub issue #12, "Reproducing NeurIPS paper Fig. 6 results: predictions have no longitudinal dimension"," Comment on Issue #12 in *ClimateSet* GitHub repository, 2024, accessed: 10 June 2026. [Online]. Available: https://github.com/RolnickLab/ClimateSet/issues/12/

A

Feature maps

The figures in this appendix show internal feature maps from the final PEAR architecture immediately before the patch recovery stage. The location of this stage within the architecture, together with the corresponding tensor shapes, is shown in Figure 3.3 and Table 3.2. At this stage, the representation contains 96 feature maps in total: 48 from the skip connection and 48 from the bottleneck pathway. The two groups are visualized separately to examine whether the skip connection and bottleneck pathways give rise to different patterns in the final representation.

Since PEAR is based on the Swin Transformer architecture, information should be able to propagate across local attention windows through the shifted-window mechanism and the hierarchical structure of the model. The original PEAR implementation was designed and validated for this receptive-field behavior at $N_{\text{side}} = 64$. However, the adapted model used in this work operates at $N_{\text{side}} = 32$, and this change in grid resolution could introduce unforeseen artifacts in the internal representations.

The feature maps shown in this appendix correspond to the best epoch of the final PEAR model configuration for three climate models where PEAR perform worse than the U-Net baseline for near-surface air temperature: FGOALS-f3-L, CAS-ESM2-0, and NorESM2-LM. Since near-surface air temperature is expected to benefit from a wide effective receptive field, these cases provide useful diagnostics for possible limitations in spatial information propagation. The skip connection and bottleneck feature maps are shown in Figures A.1 and A.2 for FGOALS-f3-L, Figures A.3 and A.4 for CAS-ESM2-0, and Figures A.5 and A.6 for NorESM2-LM, respectively.

The skip connection feature maps show the clearest window-aligned patterns, with several feature maps exhibiting sharp boundaries rather than smoothly varying spatial fields; this is especially noticeable in Figure A.3. These boundaries resemble the attention-window structure of PEAR suggesting that information may not propagate between windows as effectively as intended throughout the architecture. The bottleneck feature maps are generally smoother and less dominated by large window-like blocks, although similar patterns are still visible in some features, for example in feature 53 and 68 in Figure A.6. Overall, the visualizations indicate that the adapted PEAR architecture may be affected by window-boundary artifacts, particularly in the skip connection representations.

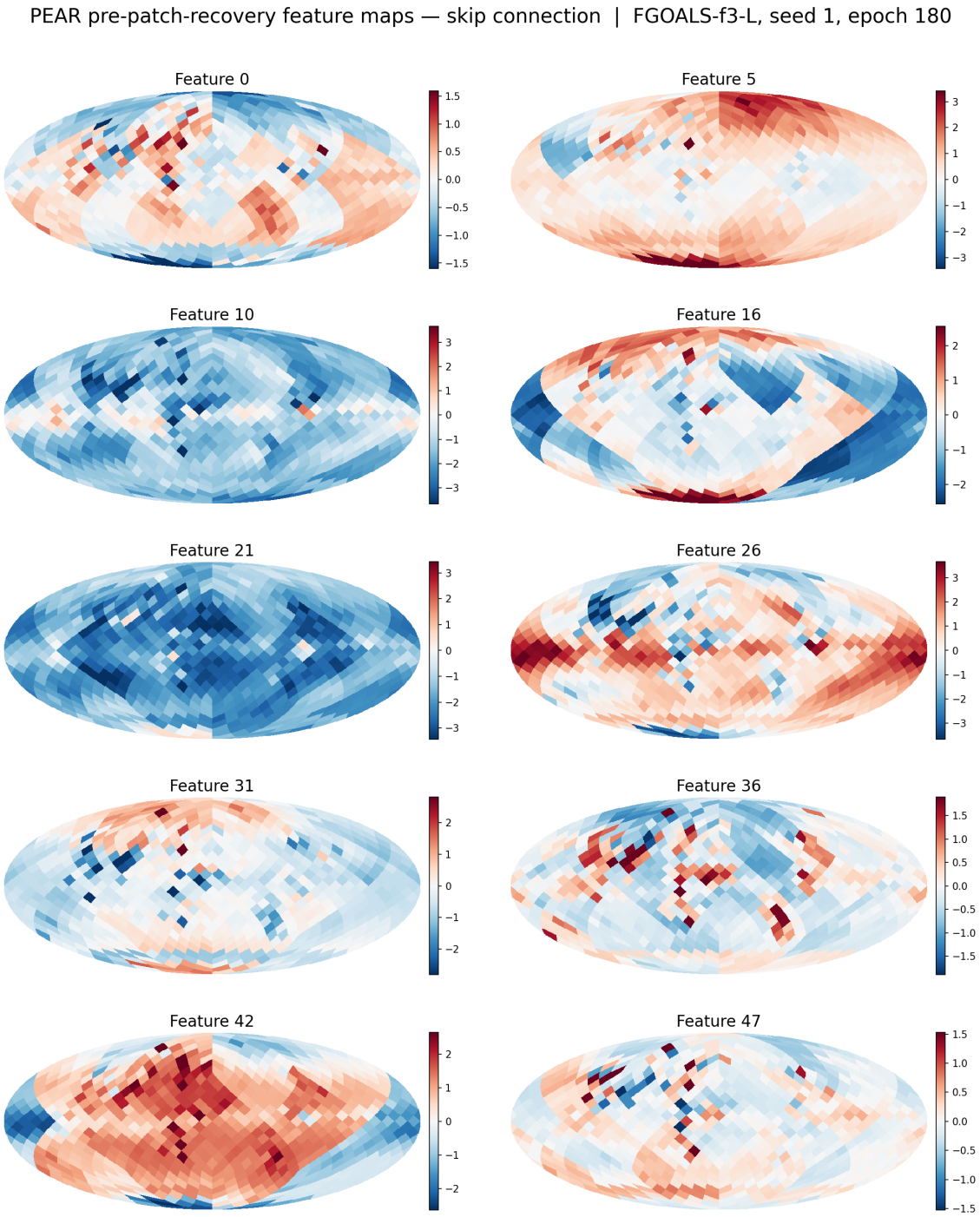

Figure A.1: Pre-patch-recovery feature maps from the final PEAR model configuration for FGOALS-f3-L. The figure shows 10 selected skip connection feature maps from the best-performing checkpoint, corresponding to epoch 180 of the seed 1 training run. The color bars show the learned feature values at each HEALPix grid cell.

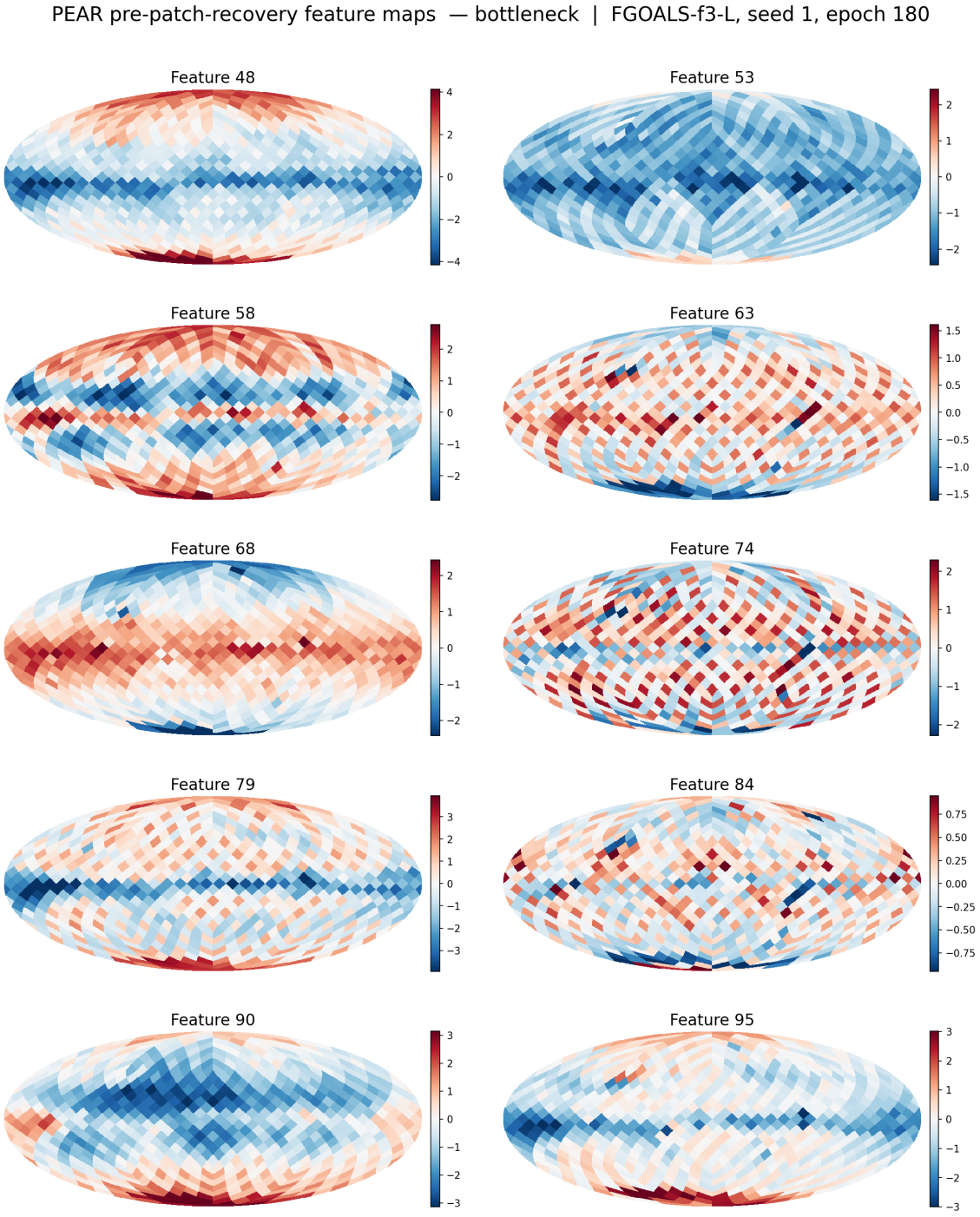

Figure A.2: Pre-patch-recovery feature maps from the final PEAR model configuration for FGOALS-f3-L. The figure shows 10 selected bottleneck feature maps from the best-performing checkpoint, corresponding to epoch 180 of the seed 1 training run. The color bars show the learned feature values at each HEALPix grid cell.

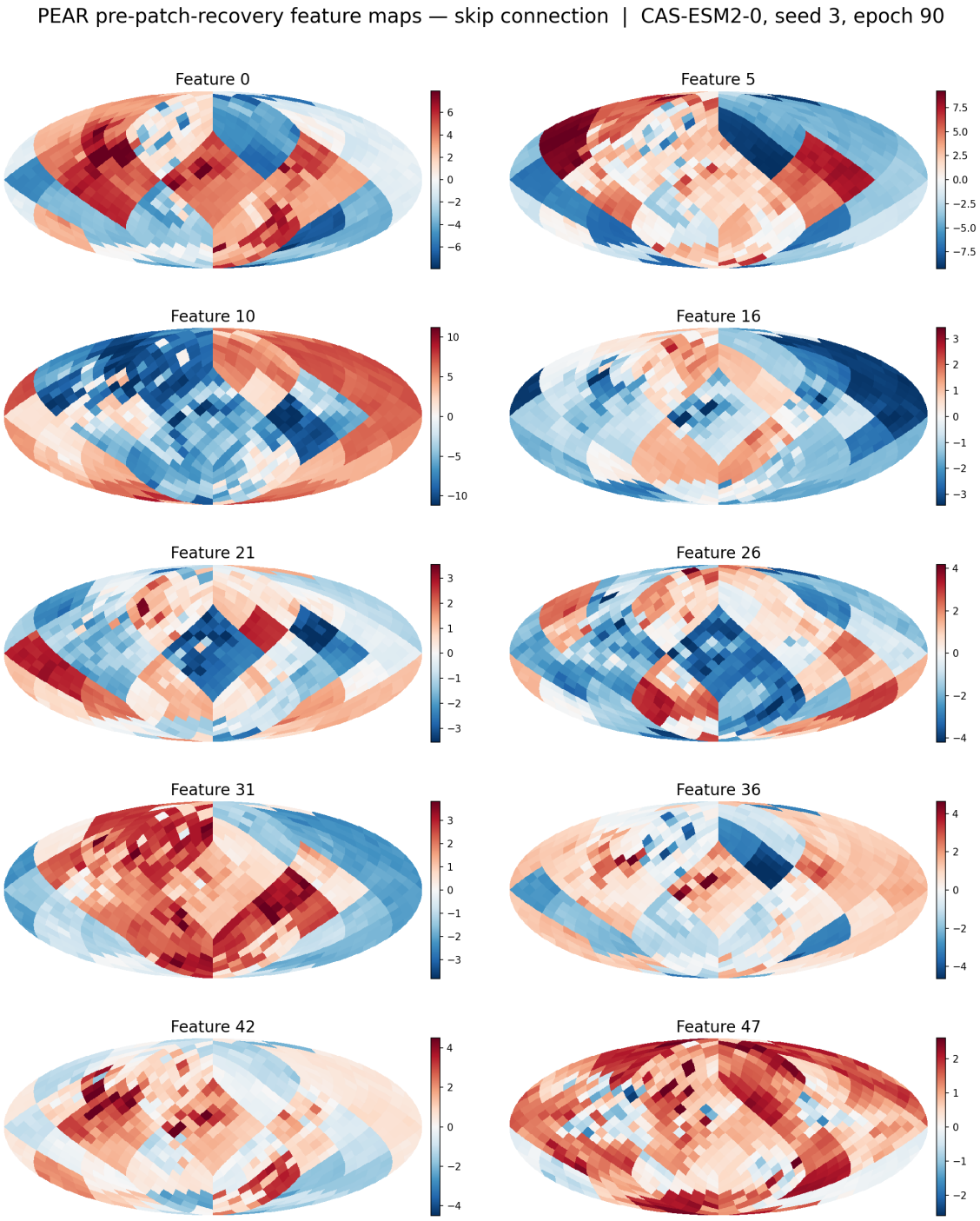

Figure A.3: Pre-patch-recovery feature maps from the final PEAR model configuration for CAS-ESM2-0. The figure shows 10 selected skip connection feature maps from the best-performing checkpoint, corresponding to epoch 90 of the seed 3 training run. The color bars show the learned feature values at each HEALPix grid cell.

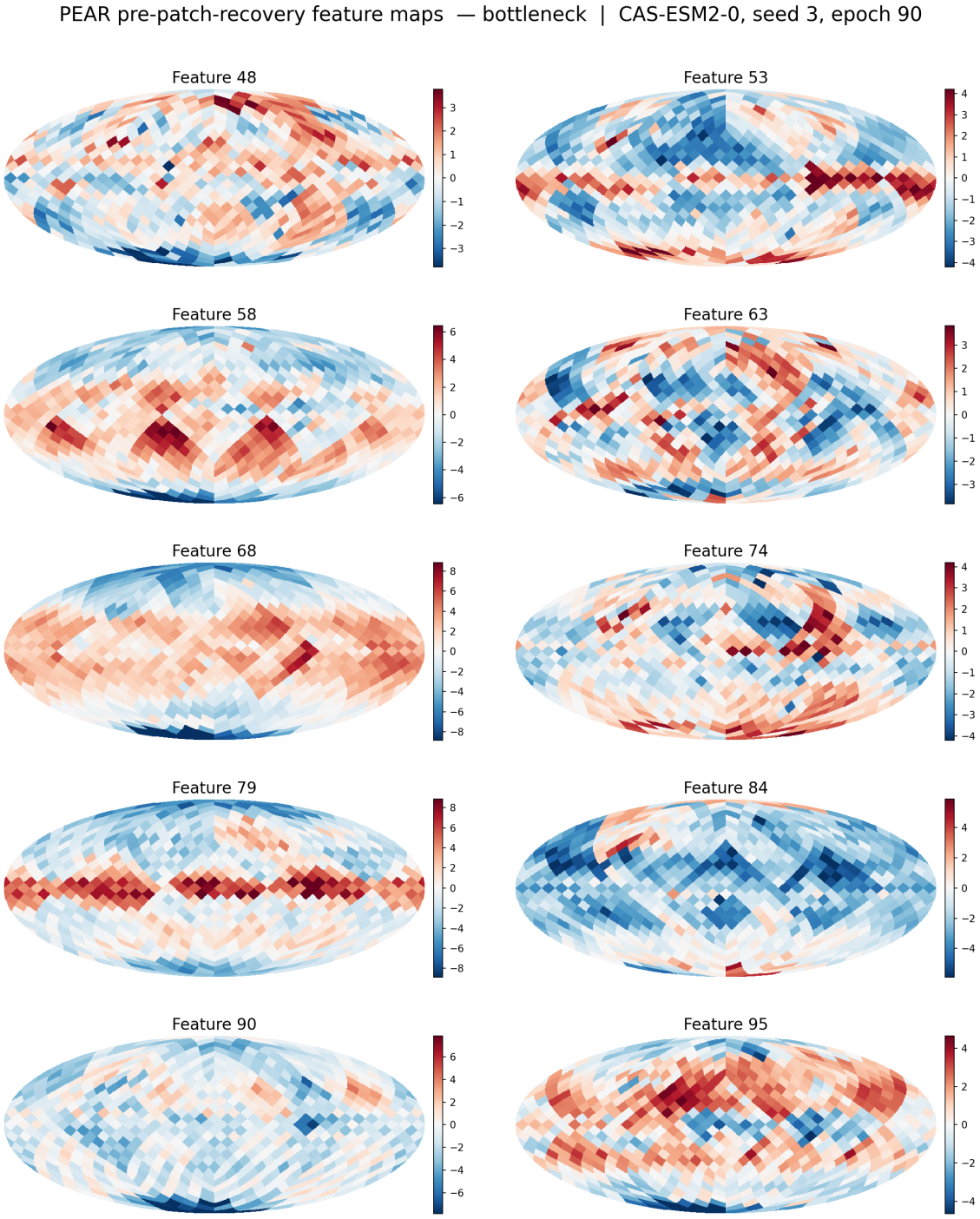

Figure A.4: Pre-patch-recovery feature maps from the final PEAR model configuration for CAS-ESM2-0. The figure shows 10 selected bottleneck feature maps from the best-performing checkpoint, corresponding to epoch 90 of the seed 3 training run. The color bars show the learned feature values at each HEALPix grid cell.

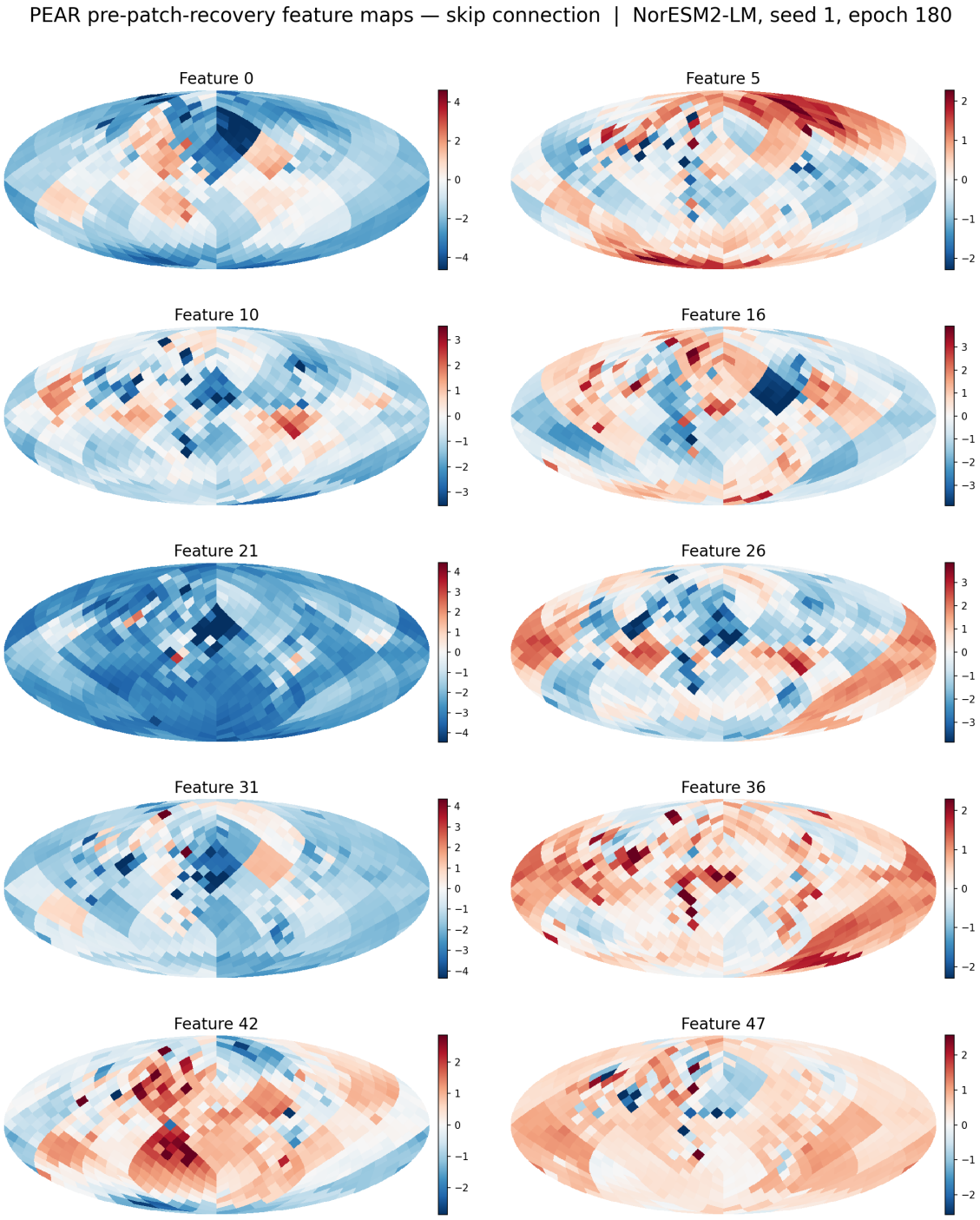

Figure A.5: Pre-patch-recovery feature maps from the final PEAR model configuration for NorESM2-LM. The figure shows 10 selected skip connection feature maps from the best-performing checkpoint, corresponding to epoch 180 of the seed 1 training run. The color bars show the learned feature values at each HEALPix grid cell.

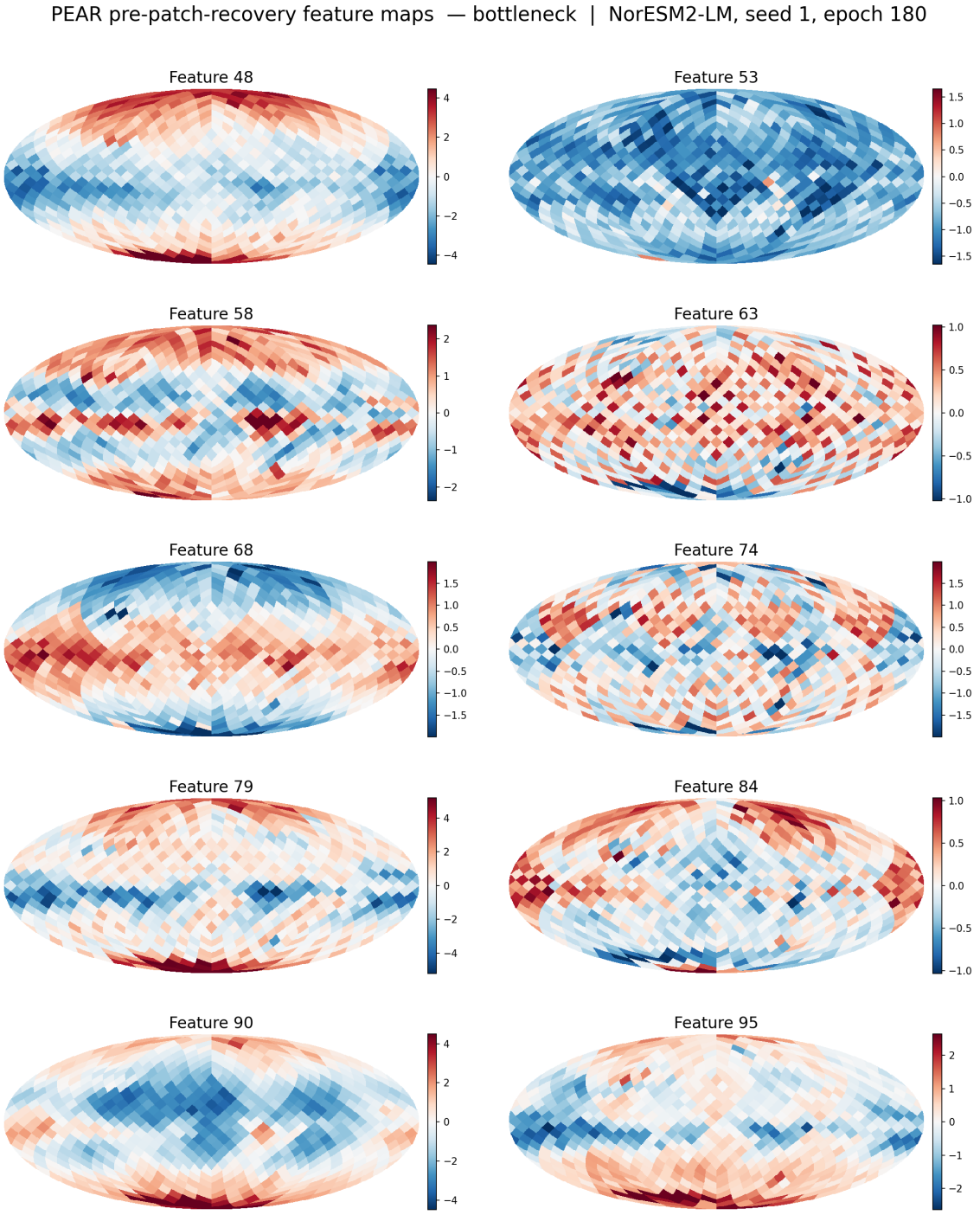

Figure A.6: Pre-patch-recovery feature maps from the final PEAR model configuration for NorESM2-LM. The figure shows 10 selected bottleneck feature maps from the best-performing checkpoint, corresponding to epoch 180 of the seed 1 training run. The color bars show the learned feature values at each HEALPix grid cell.

DEPARTMENT OF MATHEMATICAL SCIENCES
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY