Deep Learning for Empirical Downscaling of Earth Science gridded data

Carlos Alberto Gómez Gonzalez

ESA-ECMWF Workshop on Machine Learning for Earth Observation and Prediction, 16-11-2022

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

STARS POST-DOCTORAL PROGRAMME

Overview

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

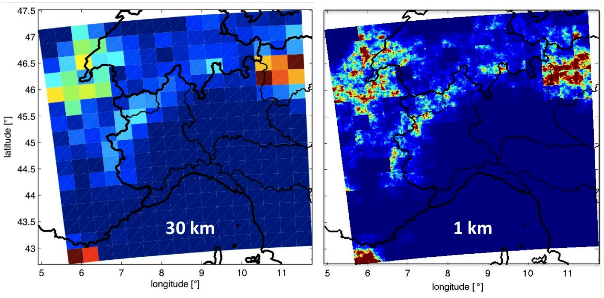
- Introduction and motivation
- Deep Learning for super-resolution and empirical downscaling
- DL4DS library
- Applications of DL4DS
- Next steps

Introduction

AI for Earth Sciences

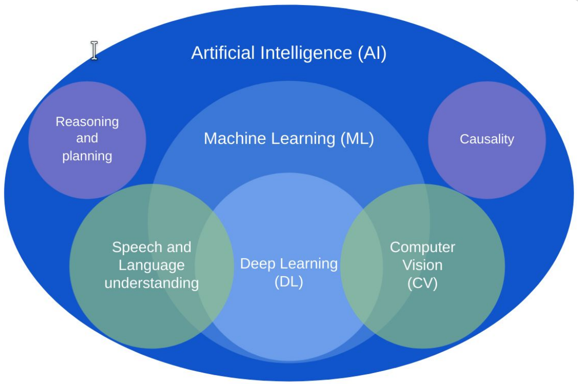
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Earth Sciences



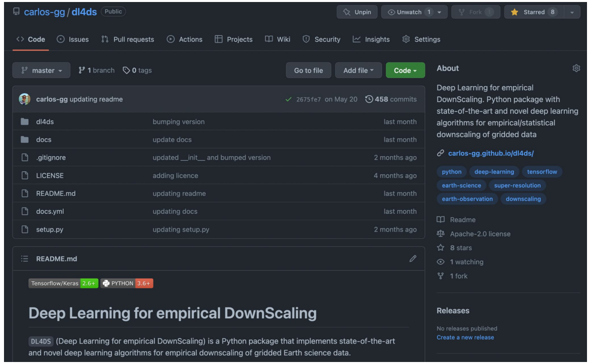
- Problem definition
- Domain expertise
- Data sources
- Baseline approaches
- Validation metrics

Artificial Intelligence



- Framing Earth science problems from a ML point of view
- Identification and development of ML methods for ES needs

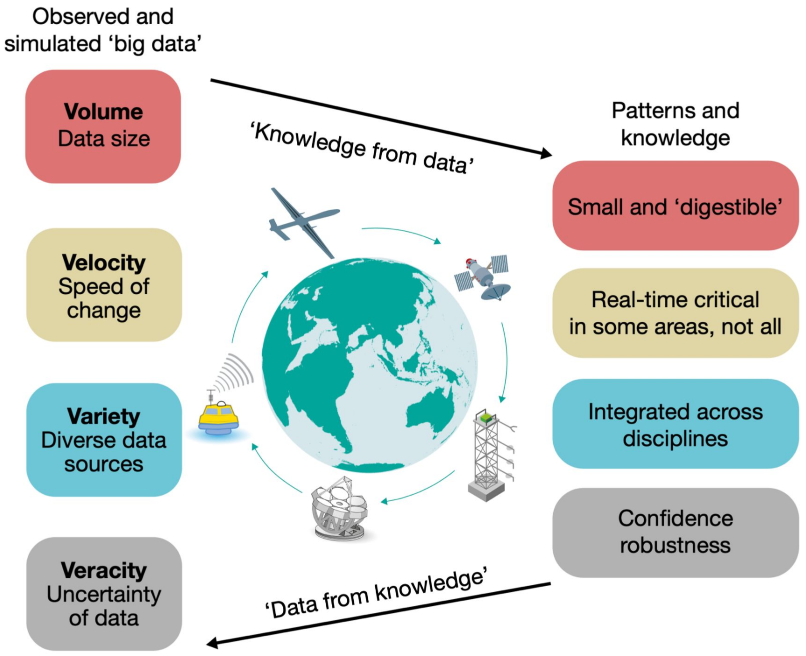
AI/ML engineering



- Development of robust, efficient and open code
- Smart testing and model design/tuning
- Reproducibility
- Scalability (HPC-ready)

AI for Earth Sciences

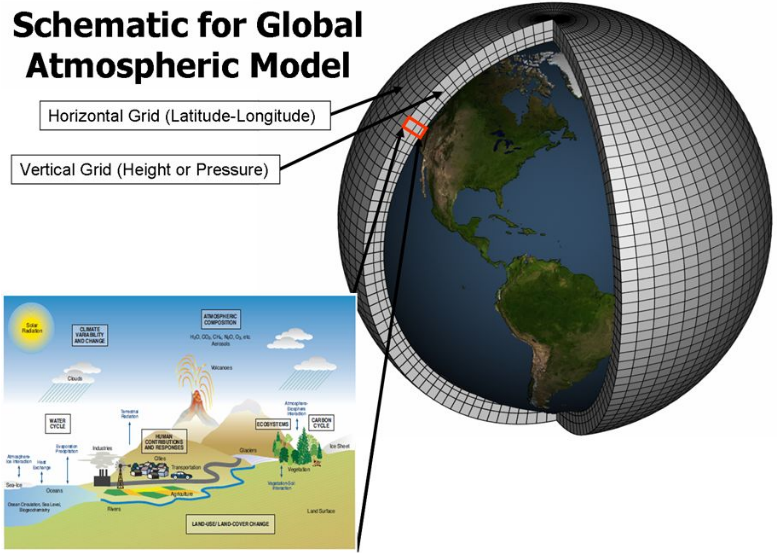
- Common tasks between AI or Computer Vision and Earth Sciences:
  - Time series forecasting → regression
  - Next frame video prediction → weather forecasting (nowcasting)
  - Super-resolution → empirical downscaling
  - Object recognition → pattern finding and detection
  - Inpainting → missing data filling
  - Image to image (domain) translation → transfer functions, surrogate models



Reichstein et al. 2019

Climate models

- Climate models use equations to represent the processes and interactions that drive the Earth's climate
    - atmosphere, oceans, land and ice-covered regions
- Based on Fundamental physical principles, that is the laws and equations that underpin scientists' understanding of the physical, chemical and biological mechanisms going on in the Earth system
    - E.g., Navier-Stokes equations of fluid motion, laws of thermodynamics, etc
- These equations are solved "numerically" in the model, which means they are approximated



$$\rho\left[\frac{\partial u}{\partial t}+\frac{\partial u}{\partial x}u+\frac{\partial u}{\partial y}v+\frac{\partial u}{\partial z}w\right]=-\frac{\partial p}{\partial x}+\mu\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}+\frac{\partial^2 u}{\partial z^2}\right)+\rho g_x$$

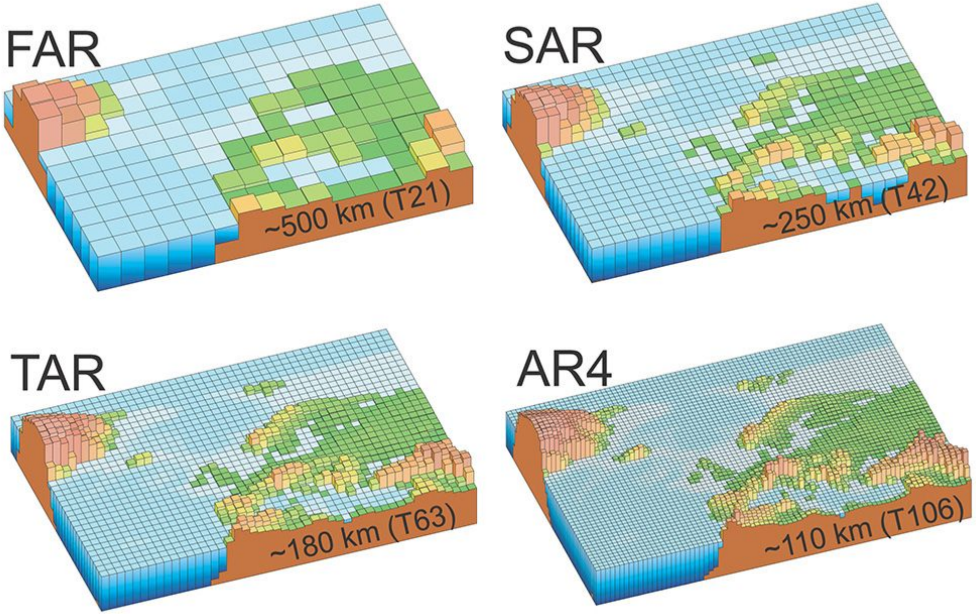
$$\rho\left[\frac{\partial v}{\partial t}+\frac{\partial v}{\partial x}u+\frac{\partial v}{\partial y}v+\frac{\partial v}{\partial z}w\right]=-\frac{\partial p}{\partial y}+\mu\left(\frac{\partial^2 v}{\partial x^2}+\frac{\partial^2 v}{\partial y^2}+\frac{\partial^2 v}{\partial z^2}\right)+\rho g_y$$

$$\rho\left[\frac{\partial w}{\partial t}+\frac{\partial w}{\partial x}u+\frac{\partial w}{\partial y}v+\frac{\partial w}{\partial z}w\right]=-\frac{\partial p}{\partial z}+\mu\left(\frac{\partial^2 w}{\partial x^2}+\frac{\partial^2 w}{\partial y^2}+\frac{\partial^2 w}{\partial z^2}\right)+\rho g_z$$

Credits: www.carbonbrief.org

Climate models and downscaling

- The idea of downscaling is to bridge the gap between the large spatial scales repre- sented by GCMs to the smaller scales required for assessing regional climate change and its impacts
- Dynamical downscaling is very expensive
  - Increasing the spatial resolution of a model by a factor of two will require ~10x more computing power



Increasing spatial resolution of climate models used through the first four IPCC assessment reports: 1990, 1995, 2001, 2007 (Credits: www.carbonbrief.org)

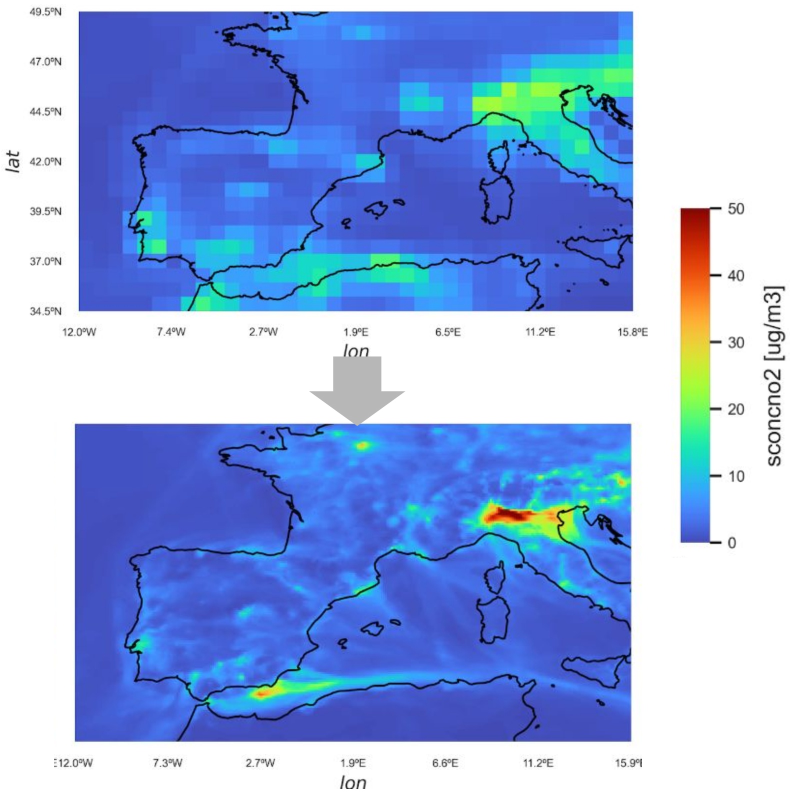
Motivation

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

- Resolution in EO depends on the satellite orbit configuration and sensor design while for ES dynamical models is a matter of computational budget
- Having more resolution (giving local insights) is important for many societal applications
- Statistical downscaling techniques present an alternative approach for learning links between the large- and local-scale climate in a more efficient way
- It enables:
    - Integration of multiple predictors (e.g., atmospheric and auxiliary variables)
    - Data fusion (other data modalities, e.g., meteo and satellite data)

Super-resolution and statistical downscaling

- The terms "statistical downscaling" and "bias correction" are used differently in different communities and countries (Maraun and Widmann 2018)
- Different meaning depending on the field (EO, weather science, S2S, atm. composition, hidrology)
- In this presentation, we mainly deal with spatial super-resolution of gridded data (EO, weather, climate)



Gomez Gonzalez 2022

Deep Learning for super-resolution and empirical downscaling

Deep Learning (supervised fashion)

$$f : \mathcal{X} \to \mathcal{Y}, \quad (x_i, y_i)_{i=1,\dots,n}$$

Gradient descent optimization

Loss function

Regularization

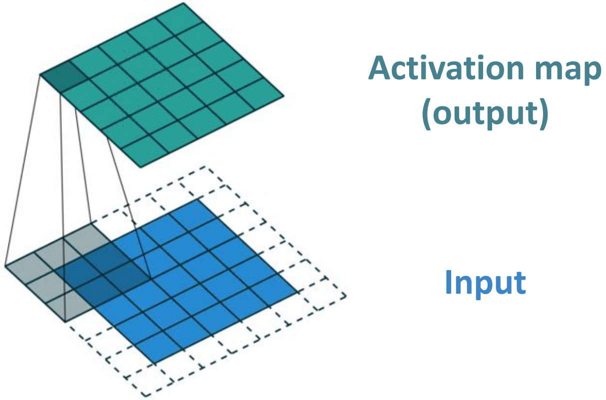
$$f = \arg\min_{f_\theta, \theta \in \Theta} \sum_{i=1}^{n} \mathcal{L}(y_i, f_\theta(x_i)) + g(\theta)$$

Model architecture

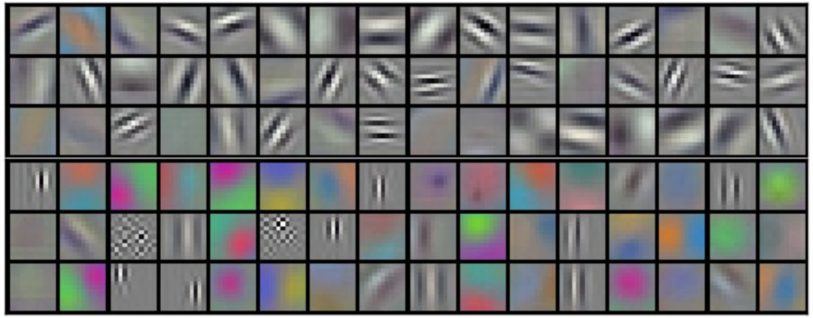
Training data

Convolutions in a nutshell

Barcelona Supercomputing Center
Centro Nacional de Supercomputación



2D convolution using a kernel size of 3 (sliding shadow) with stride of 1 and padding



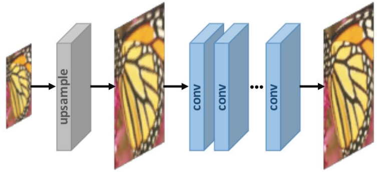
96 convolutional kernels of size 11×11×3 learned by the first convolutional layer of an image classification CNN. From Krizhevsky et al. 2012

- The convolutional layer is the core building block of a CNN and does most of the computational heavy lifting
- Its parameters consist of a set of learnable filters (see image on the top right)
- How: dot products between the entries of the filter and the input (sliding fashion)

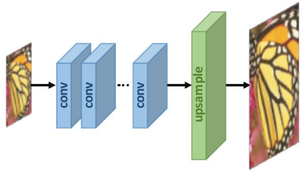
Convolution-based super-resolution

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

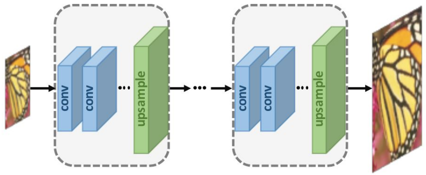
Model family



(a) Pre-upsampling SR

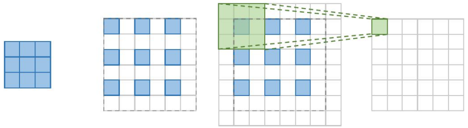


(b) Post-upsampling SR



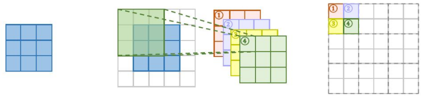
(c) Progressive upsampling SR

Upscaling methods



(a) Starting (b) Expanding (c) Convolution

Fig. 4. Transposed convolution layer. The blue boxes denote the input, and the green boxes indicate the kernel and the convolution output.



(a) Starting (b) Convolution (c) Reshaping

Fig. 5. Sub-pixel layer. The blue boxes denote the input, and the boxes with other colors indicate different convolution operations and different output feature maps.

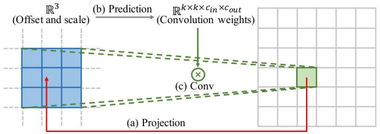
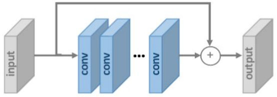
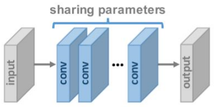


Fig. 6. Meta upscale module. The blue boxes denote the projection patch, and the green boxes and lines indicate the convolution operation with predicted weights.

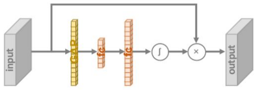
Model architectures



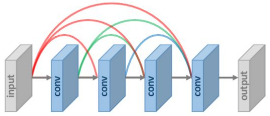
(a) Residual Learning



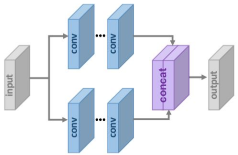
(b) Recursive learning



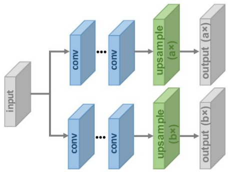
(c) Channel attention



(d) Dense connections



(e) Local multi-path learning

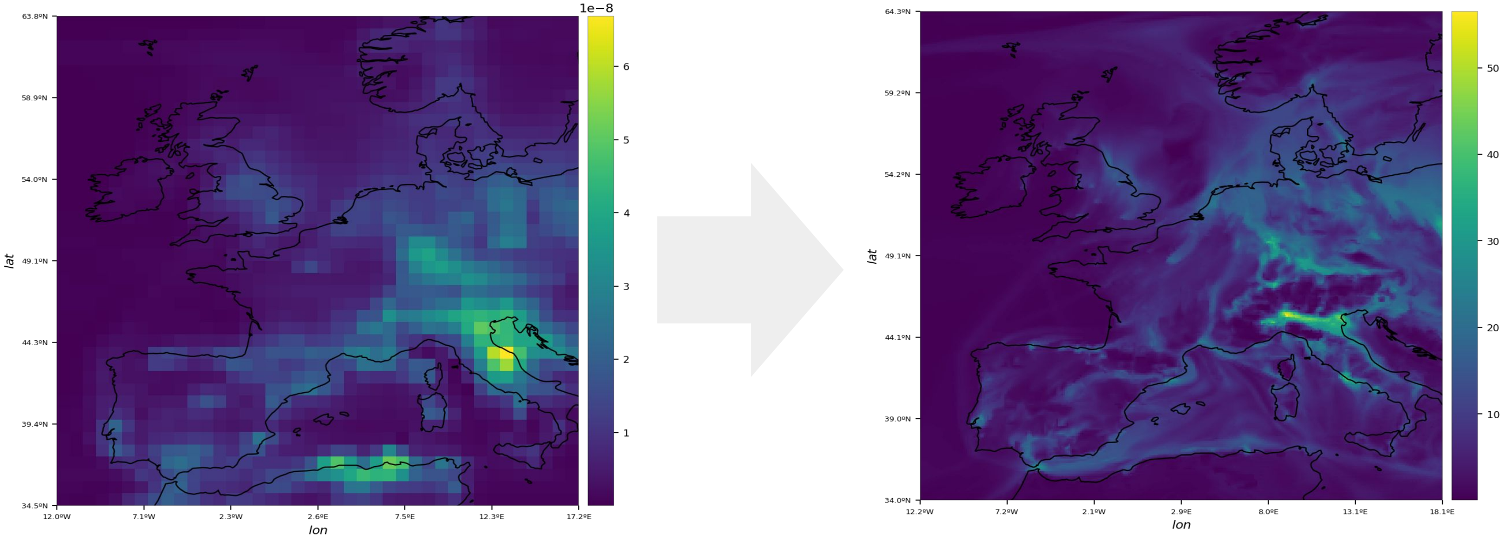


(f) Scale-specific multi-path learning

Wang, et al. 2020

DL-based super-resolution → empirical downscaling

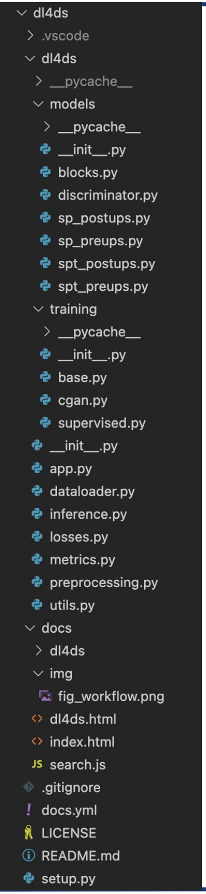
- SR ideas have inspired DL-based downscaling methods in climate science, e.g., Vandal et al. 2017, Leinonen et al. 2020, Stengel et al. 2020, Wang et al. 2021, etc
- Beware! ***Downscaling*** (climate science) == ***upscaling*** or ***super-resolution*** (computer vision), i.e., transfer from a lower- to higher-resolution grid




DL4DS library

DL4DS – Deep Learning for empirical DownScaling

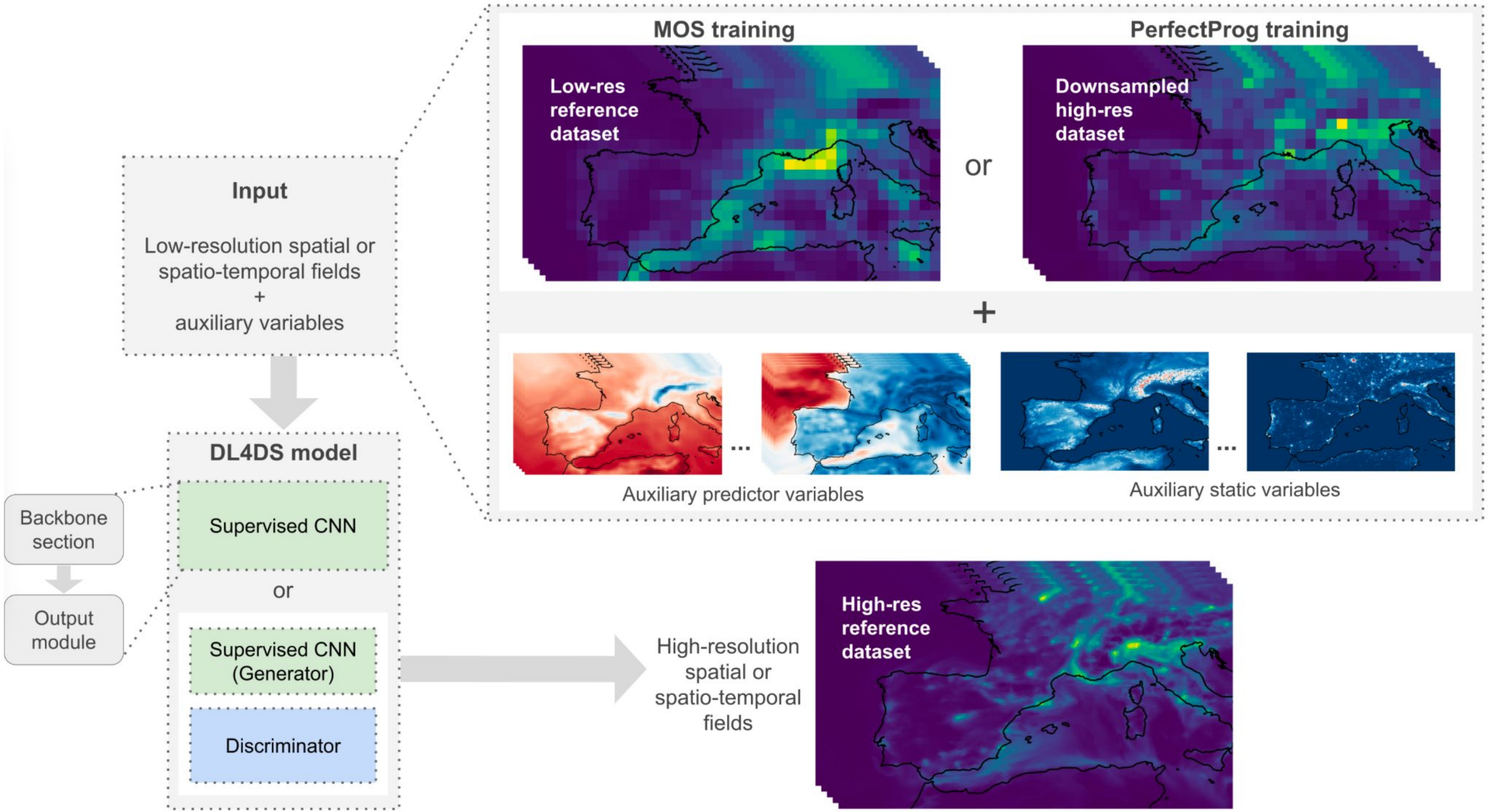
Barcelona Supercomputing Center
Centro Nacional de Supercomputación



- DL4DS - python library that implements a wide variety of state-of-the-art and novel algorithms for downscaling gridded Earth Science data with deep neural networks
- Article: “DL4DS — Deep Learning for empirical DownScaling” (*Gomez Gonzalez, in press, Environmental Data Science journal*)
- Written on top Tensorflow/Keras DL framework
  - Uses Horovod for distributed GPU training
- The models learn inter-variable spatial and spatio-temporal relationships for cross-scale translation (LowRes -> HighRes)
- These algorithms can be applied to downscale/super-resolve any gridded climate/EO dataset
- Code and tutorial: https://github.com/carlos-gg/dl4ds
- Documentation: https://carlos-gg.github.io/dl4ds/

DL4DS – Deep Learning for empirical DownScaling

Barcelona Supercomputing Center
Centro Nacional de Supercomputación



DL4DS – Deep Learning for empirical DownScaling

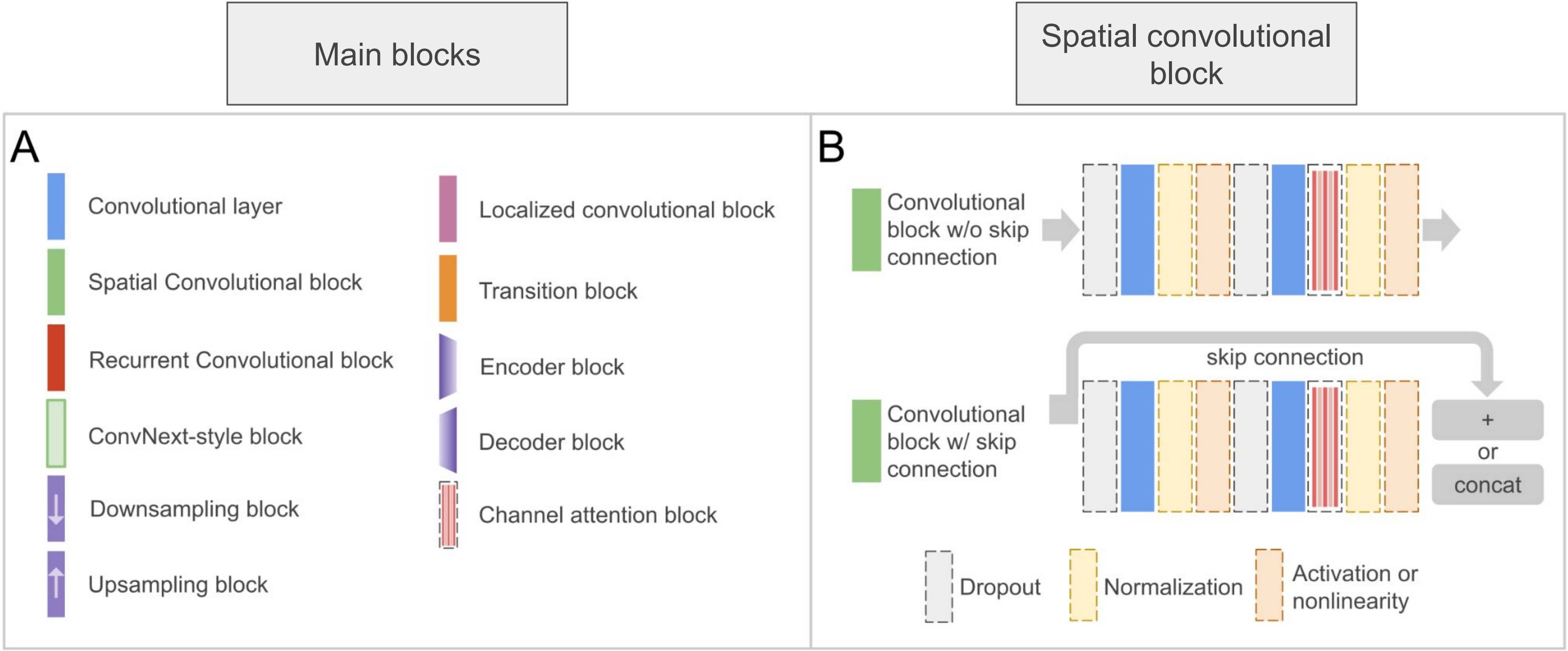
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

A wide variety of architectures are possible by mixing the following design choices:

Downscaling type	Training (loss type)	Sample type	Backbone section	Upsampling method
Explicit pairs of HR and LR datasets (MOS)	Supervised (non-adversarial)	Spatial	Plain convolutional	Pre-upsampling via interpolation
Implicit pairs, using only HR data (PerfectProg)	Conditional Adversarial	Spatio-temporal	Residual	Post-upsampling via sub-pixel convolution
			Dense	Post-upsampling via resize convolution
			Unet (pre-upsampling, spatial samples)	Post-upsampling via deconvolution
			Convnext (spatial samples)	

DL4DS building blocks

Barcelona Supercomputing Center
Centro Nacional de Supercomputación



DL4DS generators

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

- Structure of a supervised network (generator for GANs)

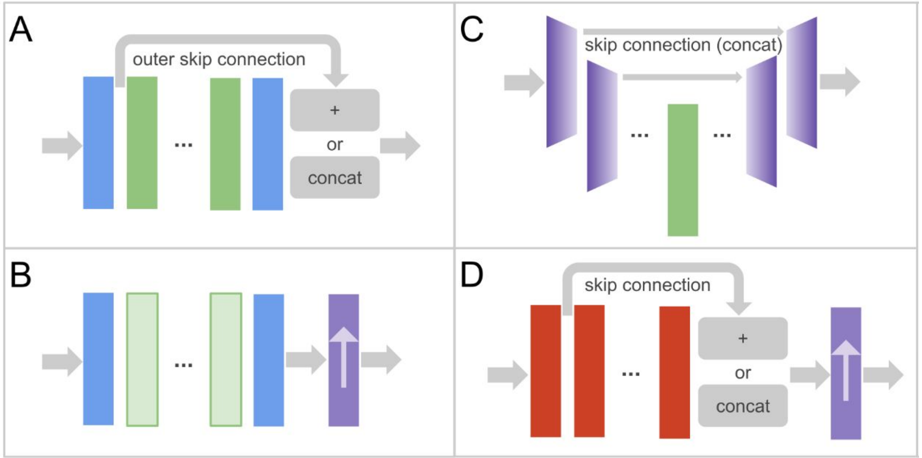
Backbone section examples → Output module

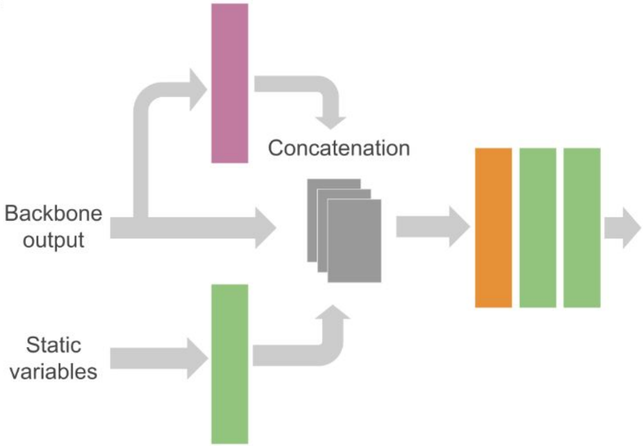
DL4DS generators

- Structure of a supervised network (generator for GANs)

Backbone section examples

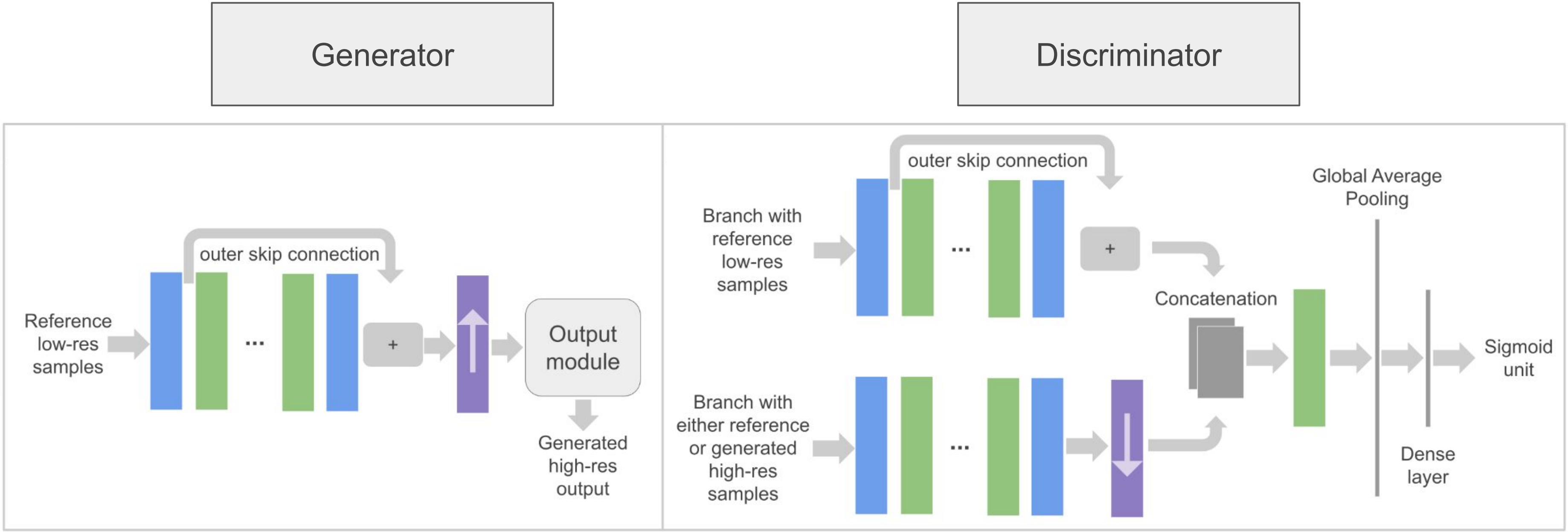
Output module





DL4DS generative adversarial models

- DL4DS allows training conditional generative adversarial models. Example:



DL4DS main classes

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

- StandardScaler, MinMaxScaler
  - Extend scikit-learn normalization classes to ND arrays
- DataGenerator (tf.keras.Sequence)
  - Returns a batch of samples (X, Y) for training
  - All the preprocessing is done here (cropping, resizing, slicing, etc)
- Trainer (SupervisedTrainer and CGANTrainer)
  - Takes care of the training procedure, feeds the networks with training samples over several epochs
  - Saves results to disk
- Predictor
  - Inference on holdout/unseen data

```
architecture_params = dict(n_filters=4, n_blocks=8, normalization='bn',
                           dropout_rate=0.2, dropout_variant='mcspatialdrop',
                           attention=False, activation='relu', localcon_layer=True)

trainer = dds.SupervisedTrainer(
    upsampling='spc', backbone='resnet',
    data_train=y_cams_train, data_val=y_cams_val, data_test=y_cams_test,
    data_train_lr=x_cams_train, data_val_lr=x_cams_val, data_test_lr=x_cams_test,
    predictors_train=[x_tas_train, x_sfcw_train],
    predictors_val=[x_tas_val, x_sfcw_val], predictors_test=[x_tas_test, x_sfcw_test],
    static_vars=[topo, laoc, urb_frac], scale=8, interpolation='inter_area',
    batch_size=32, loss=loss, epochs=100,
    device='GPU', gpu_memory_growth=True, use_multiprocessing=False,
    learning_rate=(1e-3, 1e-4), lr_decay_after=1e5,
    early_stopping=True, patience=6, min_delta=0, show_plot=True,
    save=True, save_path='./dl4ds_results', save_bestmodel=True,
    trained_model=None, trained_epochs=0, verbose=True, **architecture_params)

trainer.run()
```

DL4DS building blocks

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

```python
class ConvBlock(tf.keras.layers.Layer):
    """
    Convolutional block.

    References
    ----------
    [1] Effective and Efficient Dropout for Deep Convolutional Neural Networks:
    https://arxiv.org/abs/1904.03392
    [2] Rethinking the Usage of Batch Normalization and Dropout:
    https://arxiv.org/abs/1905.05928
    """
    def __init__(self, filters, strides=1, ks
                 activation='relu', normaliza
                 dropout_rate=0, dropout_vari
                 depthwise_separable=False, n
        super().__init__(name=name)
```

```python
class ResidualBlock(ConvBlock):
    """
    Residual block.

    References
    ----------
    [1] Deep Residual Learning for Image Recognition: https://arxiv.org/abs/1512.03385
    """
    def __init__(self, filters, strides=1, ks_cl1=(3,3), ks_cl2=(3,3),
                 activation='relu', normalization=None, attention=False,
                 dropout_rate=0, dropout_variant=None, use_1x1conv=False,
                 name=None, **conv_kwargs):
        super().__init__(filters, strides, ks_cl1, ks_cl2, activation,
                         normalization, attention, dropout_rate,
                         dropout_variant, name=name, **conv_kwargs)
```

- Blocks as tf.keras layers

DL4DS command line app

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

- DL4DS can be used not only in an interactive session but as a command line app (based on absl.flags library)
- A configuration file can be saved with the experiment parameters (excerpt shown on the right)
- HPC-friendly: Useful for running long experiments on clusters where Jupyterlab is not always available
- A Horovod call to DL4DS is shown in the example below

```
$ horovodrun -np $SLURM_NTASKS --gloo python -m dl4ds.app --flagfile=params.cfg
```

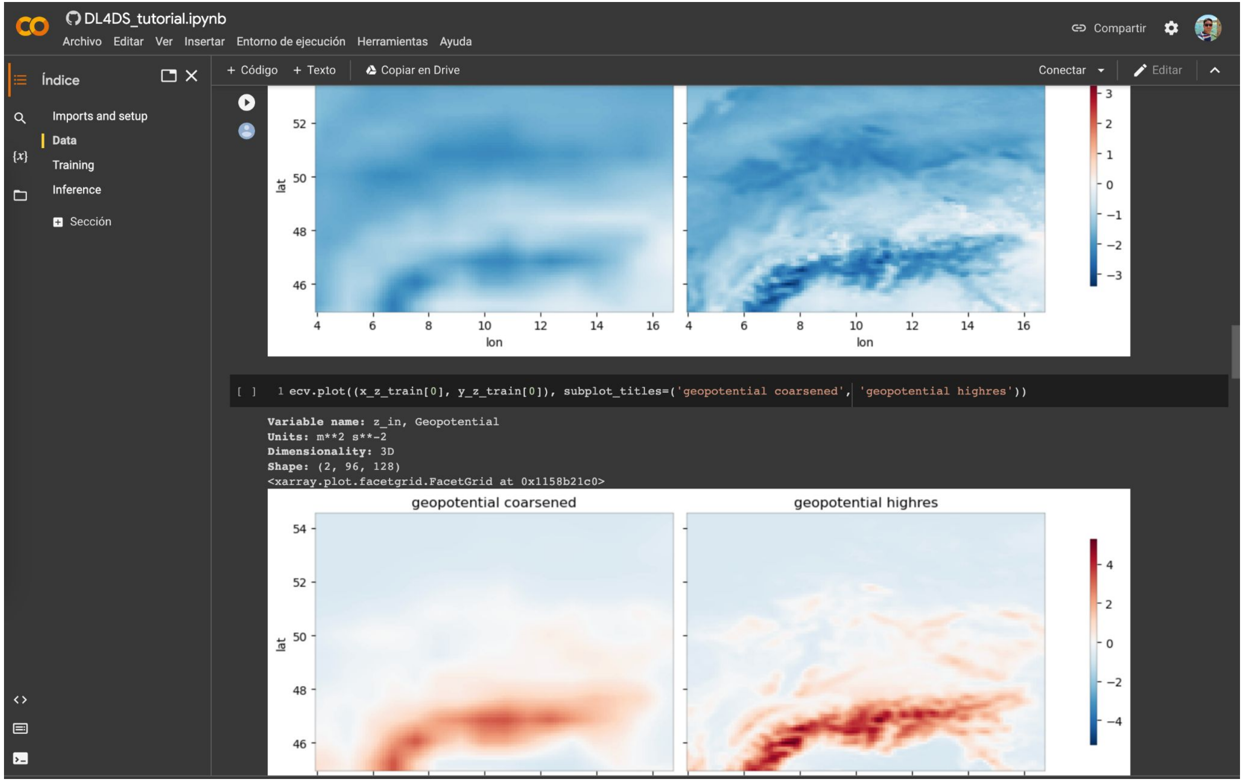
```
""" EXPERIMENT """
--data_module=data.py
--train
--test
--metrics

""" DOWNSCALING """
--trainer=SupervisedTrainer
--paired_samples=explicit
--scale=8

""" MODEL """
--backbone=resnet
--upsampling=spc
--n_filters=8
--n_blocks=10
--activation=relu
--normalization=bn
--dropout_variant=vanilla
--dropout_rate=0.2
--localcon_layer
```

Applications of DL4DS

DL4DS tutorial

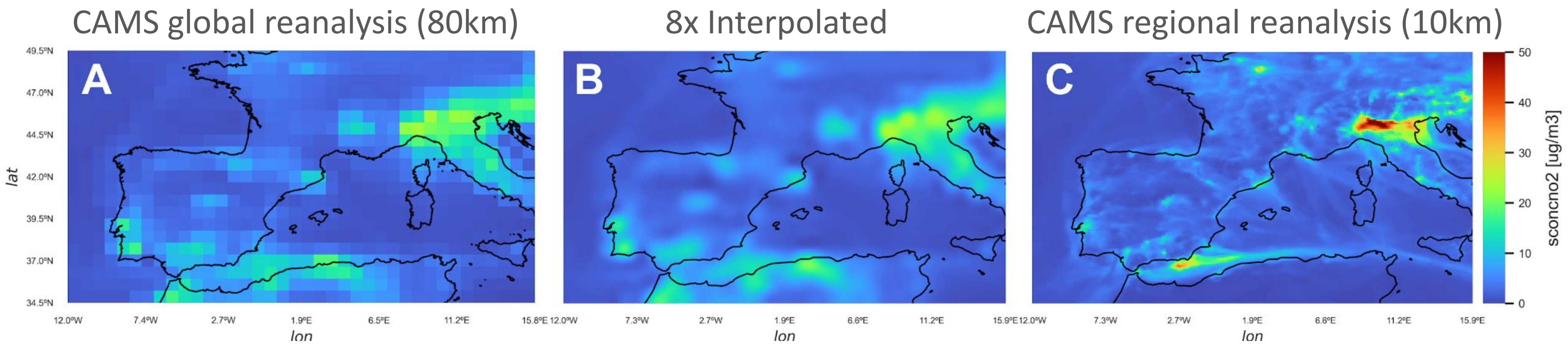
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

- Notebook tutorial available on GitHub (link to Colab)
- Application using the downscaling benchmark dataset prepared by the MAELSTROM project
- 2m temperature IFS HRES data, 8x scaling factor



NO2 surface concentration from CAMS reanalysis

Gomez Gonzalez 2022



3-hourly data from 2014 to 2018. Panel (A): reference NO2 surface concentration field from the low-resolution CAMS global reanalysis. Panel (B): resampled version, via bicubic interpolation, of the low-resolution reference field (overly smoothed and not useful). Panel (C): corresponding high-resolution field from the CAMS regional reanalysis.

Benchmark of architectures

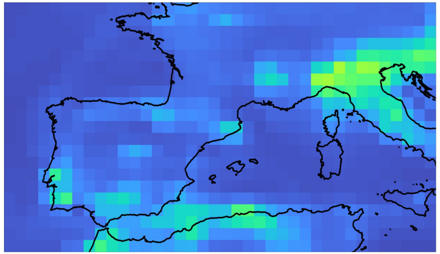
Barcelona Supercomputing Center
Centro Nacional de Supercomputación

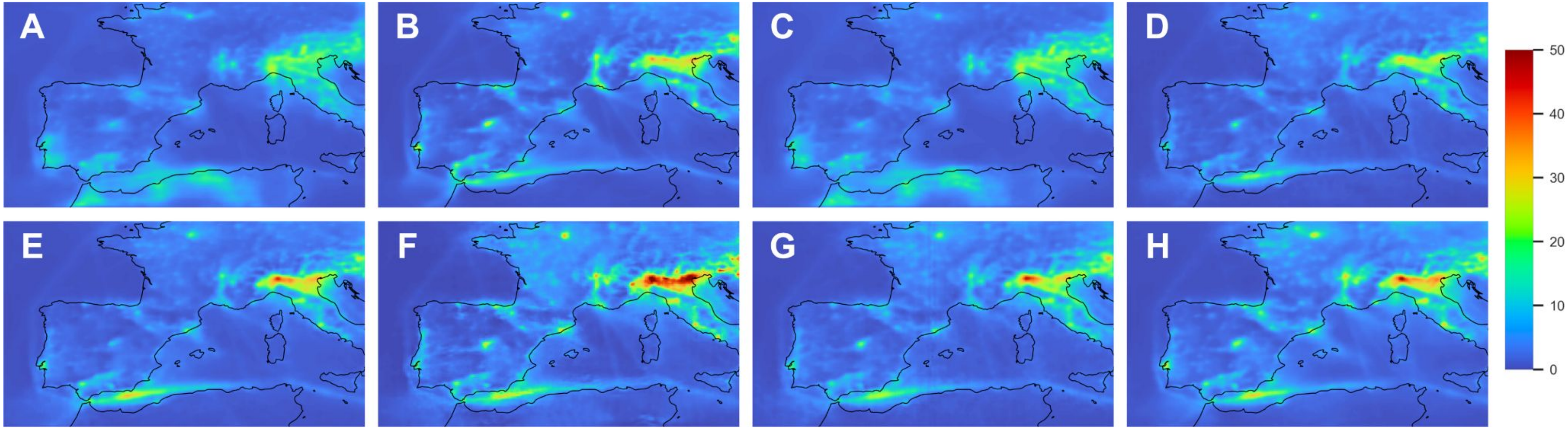
Panel	Downscaling	Learning	Sample type	Backbone	Upsampling	LCB	loss
A	PerfectProg	supervised	spatial	unet	PIN	no	mae
B	MOS	supervised	spatial	unet	PIN	no	mae
C	PerfectProg	supervised	spatial	resnet	SPC	no	dssim+mae
D	MOS	supervised	spatial	resnet	SPC	yes	dssim+mae
E	MOS	supervised	spatial	resnet	SPC	yes	mae
F	MOS	adversarial	spatial	resnet	SPC	yes	mae
G	MOS	supervised	spatiotemp	resnet	SPC	yes	dssim+mae
H	MOS	supervised	spatial	convnext	SPC	yes	mae

Without the intention of a full exploration of possible architectures and learning strategies, we chose to compare eight models trained with DL4DS. Different loss functions, backbones, learning strategies and other parameters are combined in the model architectures detailed in the table above.

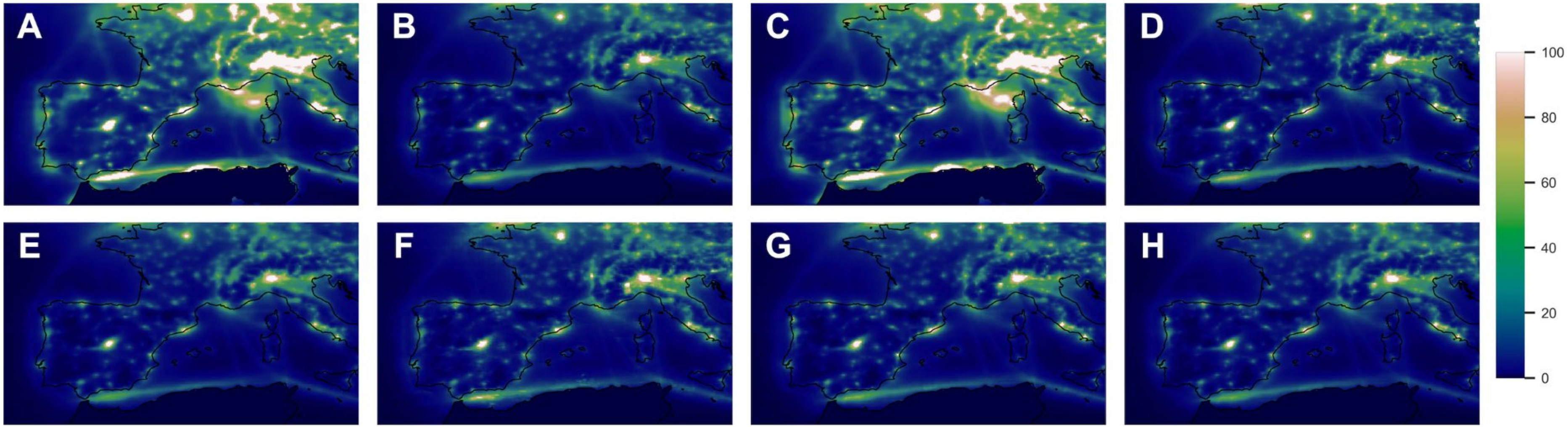
Benchmark of architectures: results

Examples of downscaled products obtained with DL4DS, corresponding to the low-resolution input grid shown to the right (for the models in the table of the previous slide).





Benchmark of architectures: results



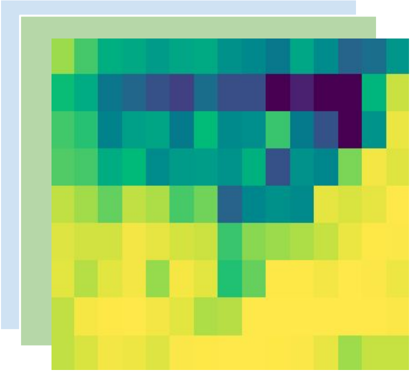
Pixel-wise RMSE for each model, computed for the whole year of 2018. The dynamic range is shared for all the panels, with a fixed maximum value to facilitate the visual comparison.

Benchmark of architectures: results

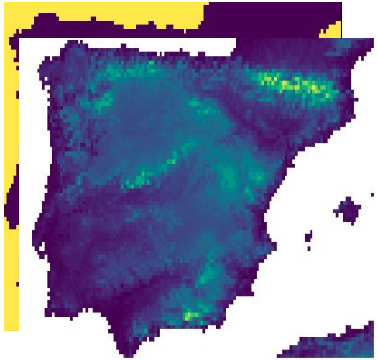
Panel	MAE	RMSE	PearCorr	SSIM	PSNR
A	2.58 ± 0.92	4.37 ± 1.50	0.64 ± 0.11	0.81 ± 0.05	32.95 ± 2.97
B	1.56 ± 0.54	2.70 ± 0.87	0.85 ± 0.04	0.89 ± 0.03	34.61 ± 2.78
C	2.58 ± 0.93	4.40 ± 1.47	0.64 ± 0.11	0.88 ± 0.04	38.85 ± 3.00
D	1.60 ± 0.60	4.75 ± 5.03	0.76 ± 0.20	**0.99** ± **0.01**	**64.87** ± **6.28**
E	**1.49** ± **0.51**	**2.56** ± **0.84**	**0.88** ± **0.03**	0.90 ± 0.03	35.23 ± 2.83
F	1.70 ± 0.58	2.87 ± 0.90	0.84 ± 0.04	0.87 ± 0.03	34.60 ± 2.81
G	1.53 ± 0.56	2.68 ± 0.96	0.86 ± 0.04	0.89 ± 0.03	34.97 ± 3.03
H	1.51 ± 0.55	2.64 ± 0.90	0.87 ± 0.03	0.90 ± 0.03	35.03 ± 2.97

- We find that models trained with explicit pairing perform better than those with implicit ones and that a supervised model with residual blocks and SPC upsampling provides the best results
- Models trained with a LCB perform better than those without it, thanks to the fact that the LCB learns grid point- or location-specific weights

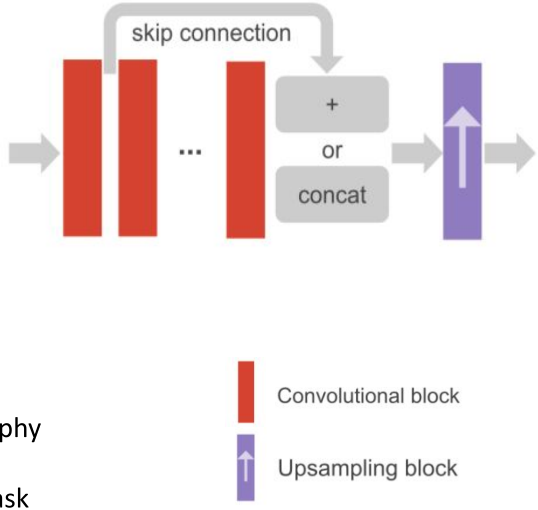
ECMWF SEAS5 downscaling of precipitation fields



Raw 1° SEAS5
daily precipitation
anomalies
+
encoded DOY

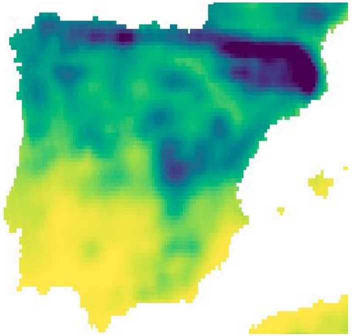


HR 0.1° topography
&
Land-ocean mask



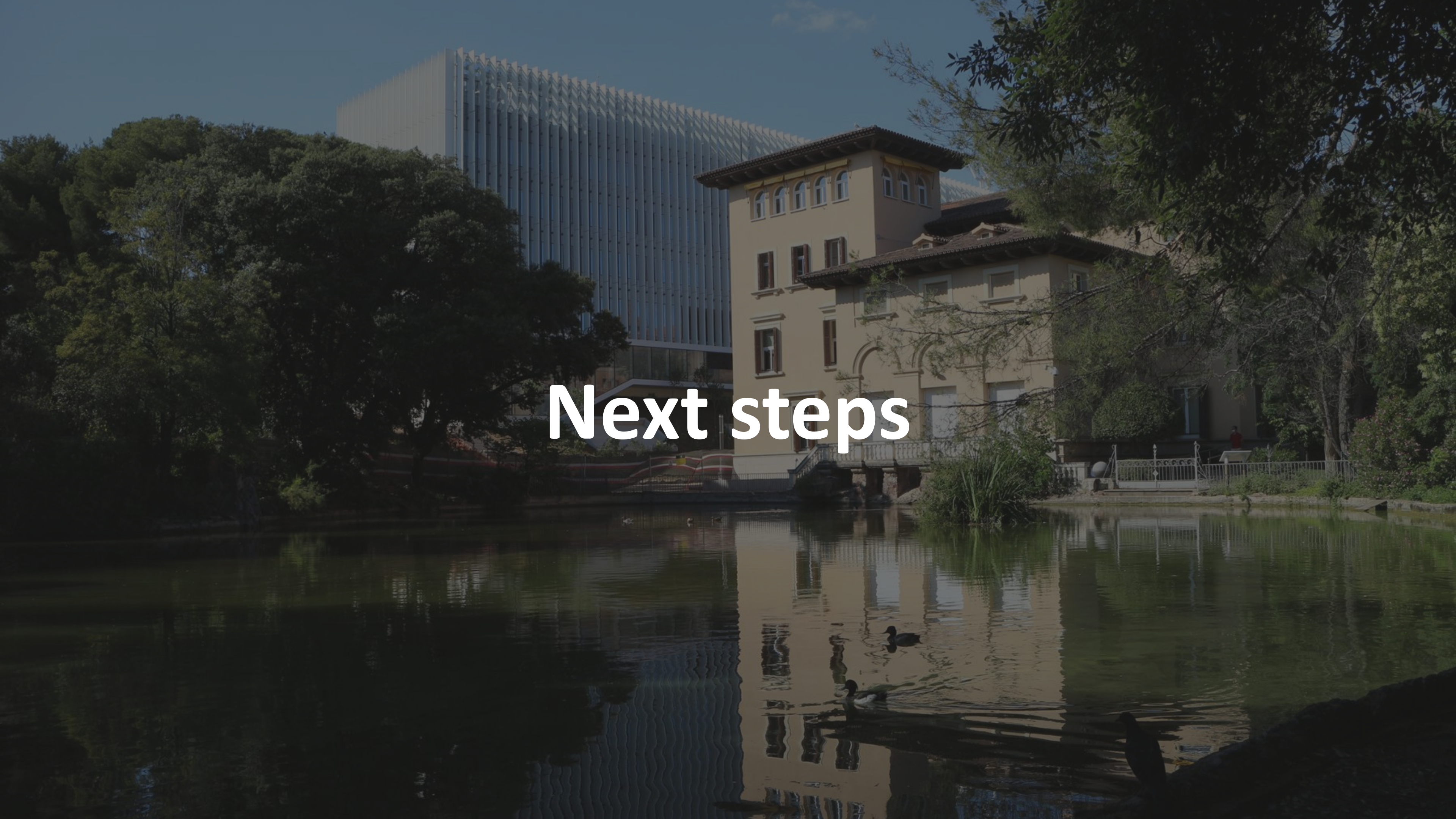
10x downscaling

HR 0.1° SEAS5 daily precipitation



Palma et al. (in-prep)

Next steps



AI and user-oriented Earth Science applications

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Data

- Benchmark data
- APIs for reanalysis, GCM/RCM, forecasts data
- EO and satellite data

Trained DL models

- Open source ecosystem of libraries for ES problems
- “Repository” for trained models (weights and specifications)

Usecase I

Usecase II

- Transfer learning
- Fine tuning
- Online learning (model update)

Next steps

- DL4DS as ML-package template for other tasks, e.g. weather forecasting
- Additional backbones
  - Transformers, diffusion models, normalizing flows, graph neural networks, Fourier neural operators
- Condition on point/station data – bias correction
  - Adding terms to the GAN loss
  - Neural processes
- Arbitrary scaling factors
  - Implicit neural representations
- Uncertainty estimation techniques
  - Monte Carlo dropout (already implemented in DL4DS)
  - Training ensembles of networks
  - Perturb the inputs (inject noise)
  - Generative modelling techniques

¡Gracias!

Acknowledgements:

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement H2020-MSCA-COFUND-2016-754433.

Barcelona Supercomputing Center
Centro Nacional de Supercomputación

Where to find me:

https://carlos-gg.github.io/

https://github.com/carlos-gg/

https://www.linkedin.com/in/carlosgog/

carlos.gomez@bsc.es