CAMAERA

Desert dust emission with neural network

Nathan Capon (HYGEOS)
Samuel Remy (HYGEOS)

PROGRAMME OF THE EUROPEAN UNION
Copernicus
ECMWF
Coordinated by HYGEOS

Task 6.1 : New desert dust emission scheme

The objective of WP6 is to develop, implement, and evaluate novel parameterizations for the emissions of online aerosols and precursor gases.

Specifically, this presentation covers the **machine learning approach for global desert dust aerosol emissions**, leveraging the best estimates produced by WP1.

Scope of this work:

- Develop a neural network-based desert dust emission scheme
- Replace the parameterized reference scheme with a data-driven ML model
- Ensure compatibility with **IFS-COMPO** for operational integration

This task relies on the results produced by Task 1 of Work Package 5 for Whitecap Fraction estimation.

Input Data Overview [Predictors]

The desert dust emission phenomenon is primarily influenced by *wind* and *soil type*. Therefore, as inputs, we selected:

Wind	Static Soil	Seasonal Soil	State Variables
Wind speed	Clay fraction	Roughness length	Soil moisture
Wind gusts	Sand fraction	Surface albedo	Snow depth
Friction velocity	Bare soil fraction	LAI (Leaf Area Index)	
Threshold velocity	Orography std. dev.		

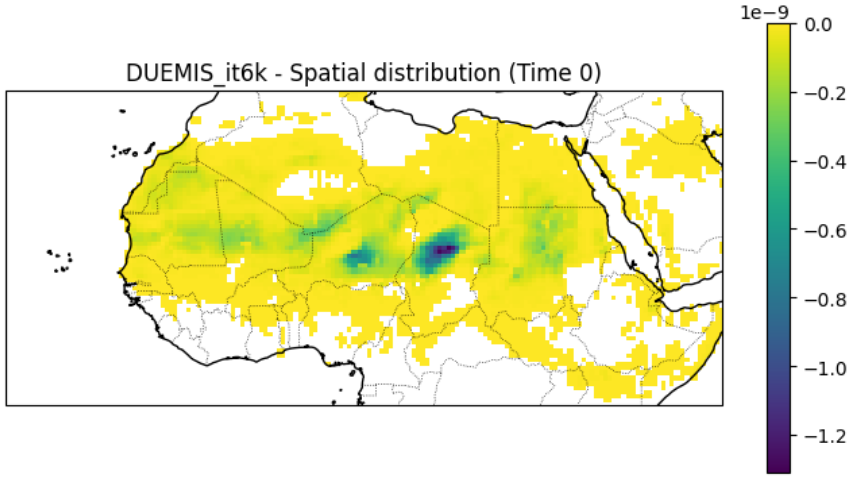
The inputs include static climatological data, annually varying features, and daily dynamic variables.

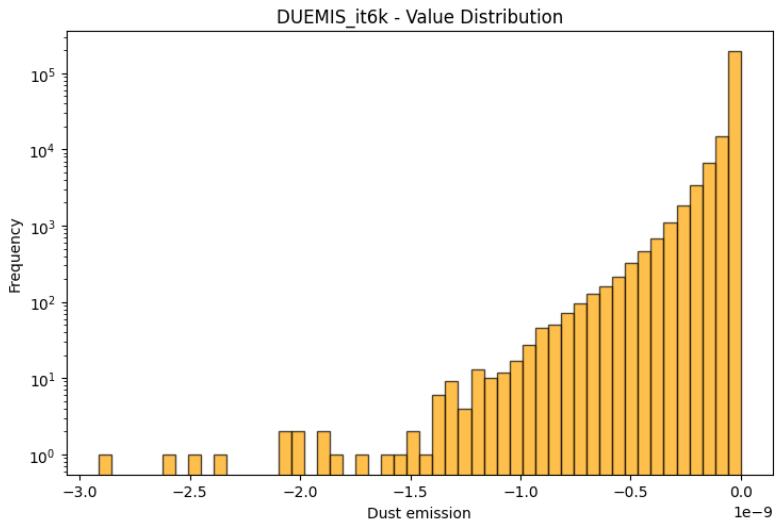
Input Data Overview [Reference]

The different schemes that can be taken as reference:

- **it57** : New Desert-Dust scheme
- **it69** : New Desert-Dust scheme + scaling factor from WP1 with AOD as control
- **it6k** : New Desert-Dust scheme + scaling factor from WP1 with DOD as control

For it6k, min = -2.92e-09 / max = -6.66e-16 / mean = -2.90e-11





Model Selection

Three key constraints guided our model choice:

Constraint	Implication	Excluded Models
IFS-COMPO parallelization	Only local pixel values accessible	CNNs, UNets, Transformers
Continuous output	Smooth predictions required	Decision trees, Random Forests, GBDT
Non-linear relationships	Complex feature interactions	Ridge, Lasso, Linear Regression

→ The most suitable model is a **Feed-Forward Neural Network** (DNN)

Remark: All models are implemented in *PyTorch* to ensure compatibility with IFS-COMPO for saving and loading.

Training Description [Data]

Temporal split (no data leakage):

- *Training:* 2018–2020 (36 monthly files) → 90% train / 10% validation
- *Test:* 2017 (12 monthly files)

Pixel-to-Pixel approach:

- Grid: 361 × 720 = *260K spatial locations* per timestep
- Retrieve valid pixels after masking zero emissions *(not so easy)* (~600K per file)

Normalisation on inputs:

- Features span very different scales (e.g., wind speed in m/s vs. clay fraction 0–1)
- *Normalisation* → Fix the magnitude discrepancy between inputs. (Prevent high-magnitude inputs from overshadowing others)

Hyperparameter Optimisation

Finding the best hyperparameters is like searching for the highest point in a mountain range **without seeing the map**.

Approach	How it works	Problem
Random Search	Sample randomly	Wastes trials on already-explored regions
Grid Search	Test every combination	Too slow (combinatorial explosion)
Bayesian Optimisation	Learn from past trials	Smart & efficient exploration

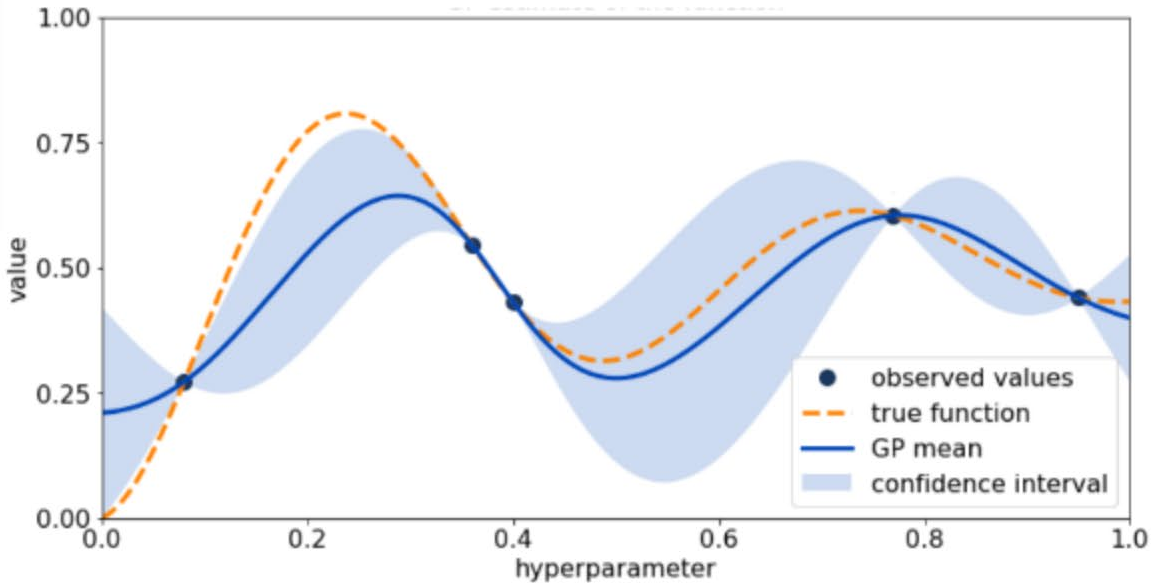
Key idea: Build a probabilistic model (surrogate) of the objective function $\rightarrow$ Use it to decide where to sample next.

Bayesian Optimisation [How It Works]

The Bayesian optimization loop :

1. **Evaluate** initial configurations
2. **Build** a surrogate model to predict performance
3. **Select** next candidate (balance *exploration* vs *exploitation*)
4. **Evaluate** → Update surrogate model
5. **Repeat** until maximum number of trials is reached

Illustration of the Bayesian optimization process



Hyperparameter Optimisation

- Implementation with **TPE sampler** (Gaussian Mixture Model) & **Median Pruner**
- **Objective**: Maximize R^2 score on the test set
- 100 trials executed, each training a model with sampled hyperparameters

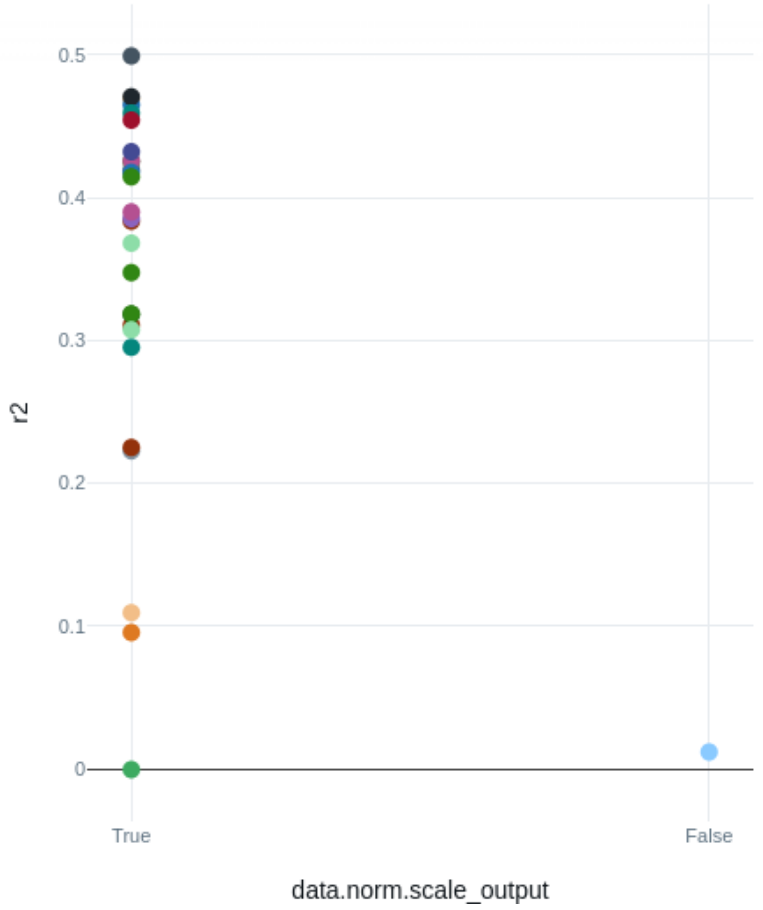
Search space (13 hyperparameters):

Category	Parameters
Architecture	Layers, Hidden dim, Activation, Dropout
Training	Batch size, LR, Weight decay, Grad clip
Optimizer	Type, Scheduler
Normalization	MinMax, Standard
Loss	MSE, MAE, R^2

Hyperparameter Optimisation [Norm]

Output normalization → faster convergence & stable gradients

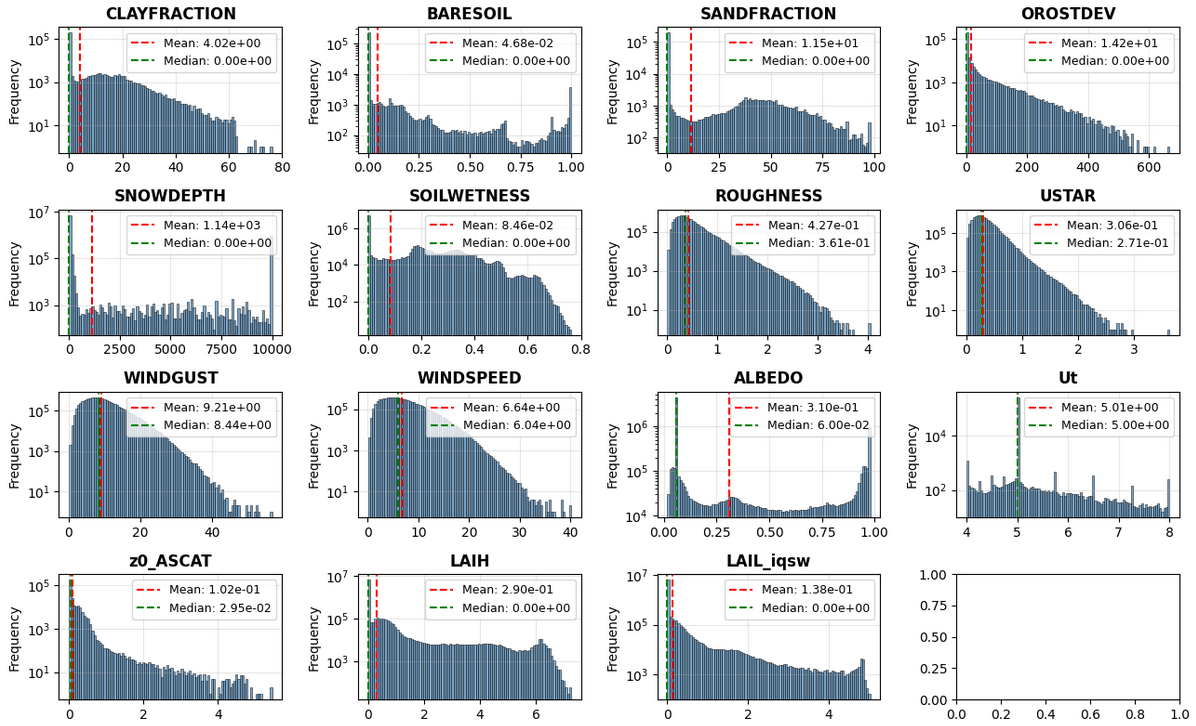
Why? The dust emission variable has a very small magnitude, which leads to the *vanishing gradient problem*. Normalizing the output prevents this issue.



Hyperparameter Optimisation [Input Norm]

Min-Max normalization preferred over Standard (z-score) normalization

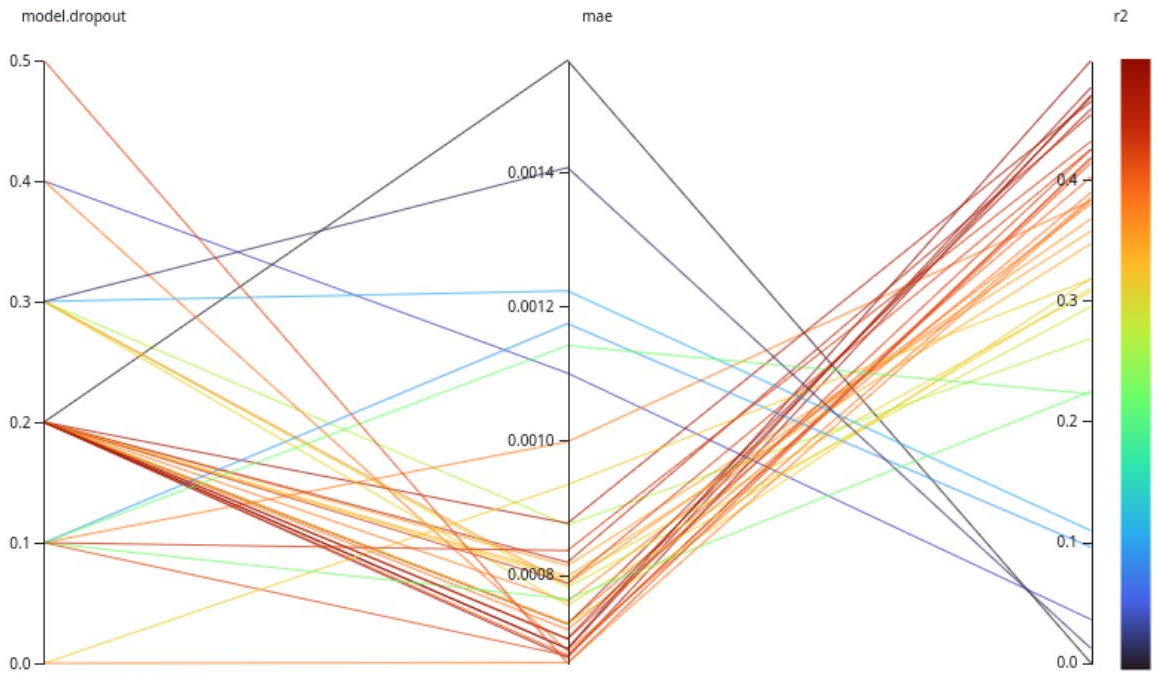
Why? Our predictors do not follow a Gaussian distribution. Applying z-score normalization would not be appropriate, making Min-Max scaling the better choice for input features.



Hyperparameter Optimisation [Dropout]

Dropout → encourages generalization

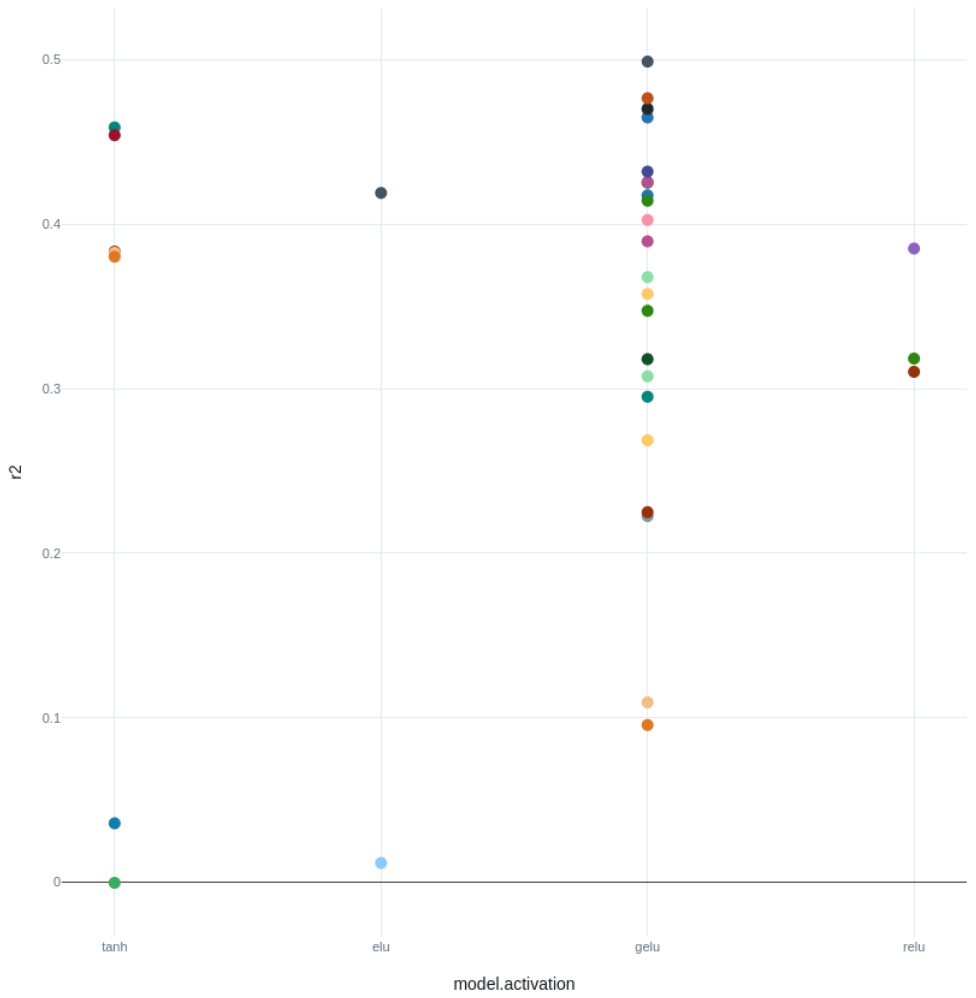
Why? By randomly deactivating neurons during training, dropout prevents the model from relying too heavily on any single neuron, forcing it to learn more robust features.



Hyperparameter Optimisation [Activation]

Activation function should be continuous.

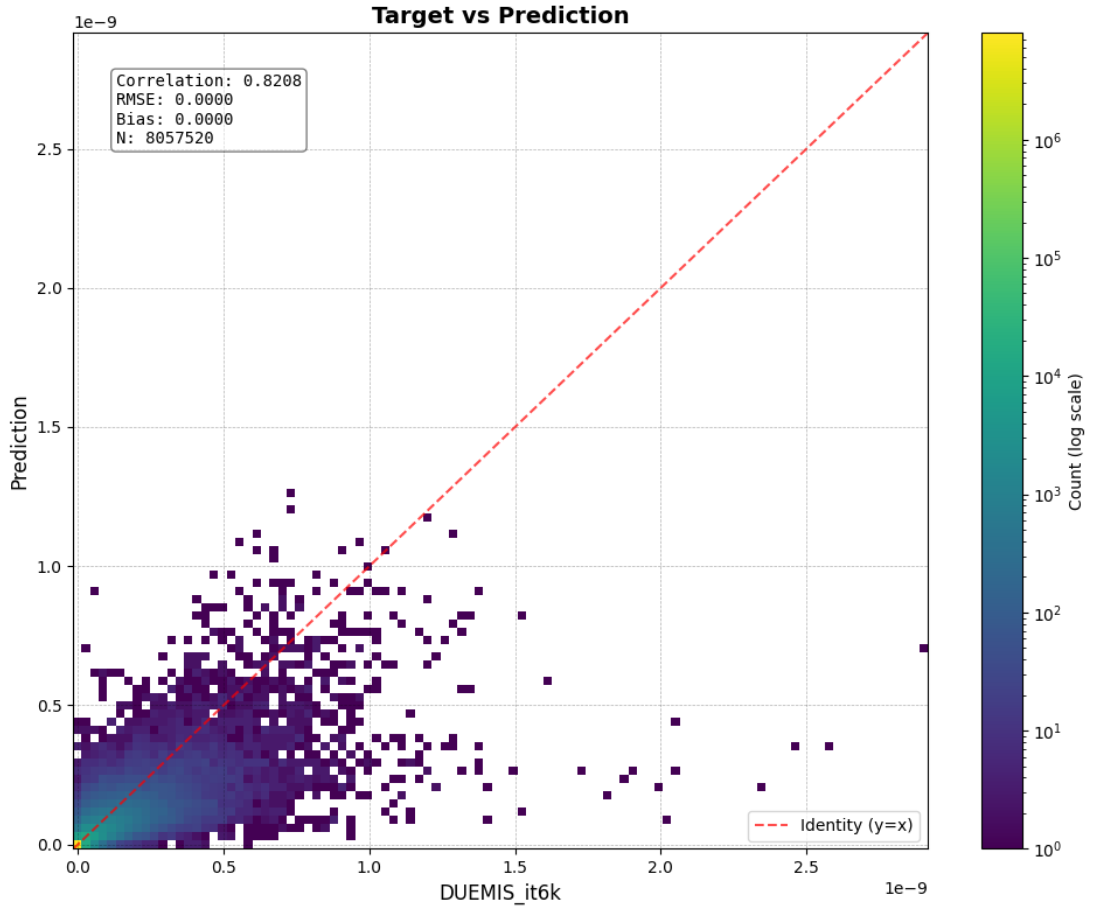
Why? Since we are predicting a physical emission scheme, preserving continuity throughout the network is essential and it also ensures stable gradient flow during training.



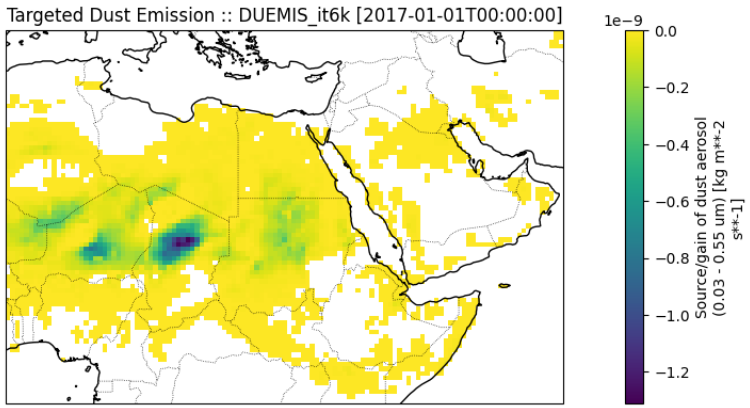
Final Model architecture

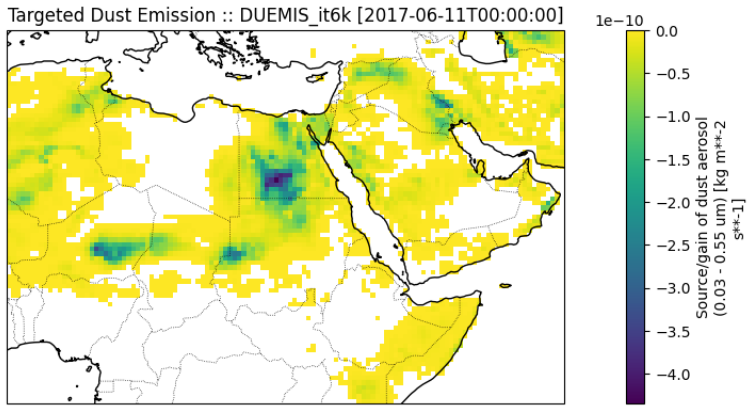
Final configuration for our DNN model:

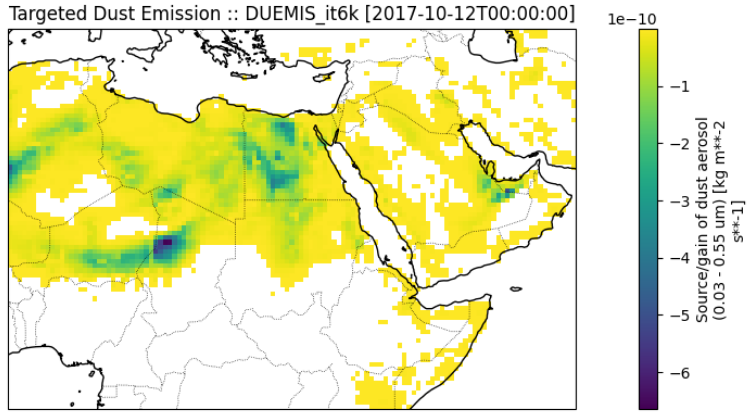
Hyperparameter	Value
Layers	[256,256,256]
Activation	GeLU
Normalisation	Inputs & Output
Dropout	0.2
Loss	r2
Optimizer	AdamW
learning rate	1e-4
batch size	256

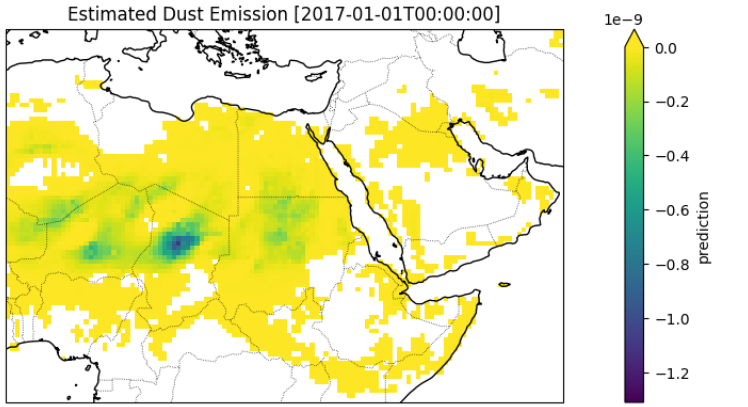


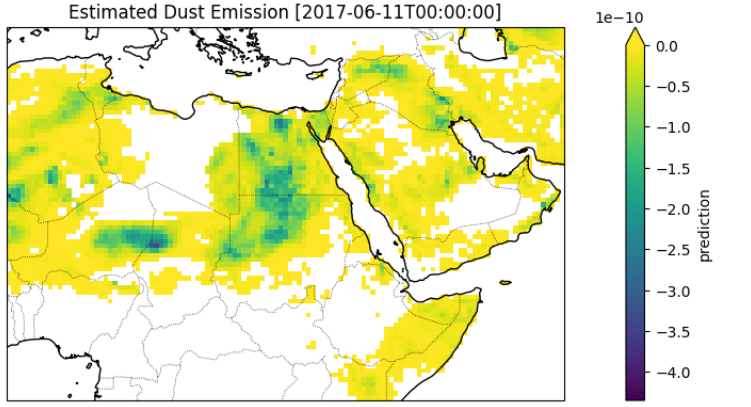
Final Model architecture

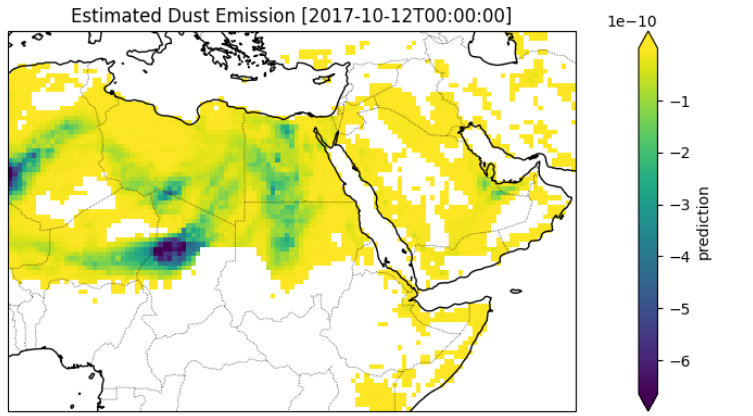












Integration into IFS-COMPO

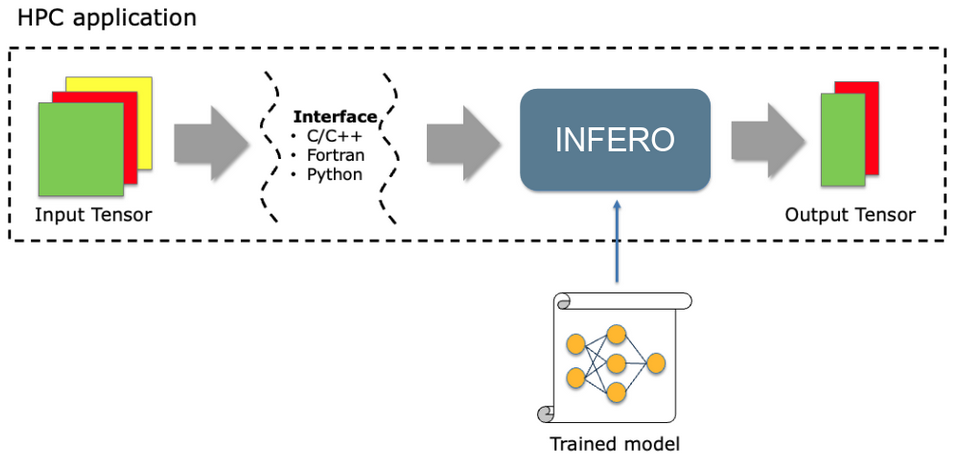
Following the approach from Task 5.1, we use the INFERO library to integrate our model into IFS-COMPO via the ONNX format.

Key modifications:

- Updated INFERO library to support **ONNX opset 25** (from opset 15)
- Normalization steps are now **embedded in the ONNX graph** → no longer applied in the Fortran wrapper

The model is successfully loaded and runs with all 15 predictors inside IFS-COMPO

Representation of the incorporation of our model into IFS-COMPO



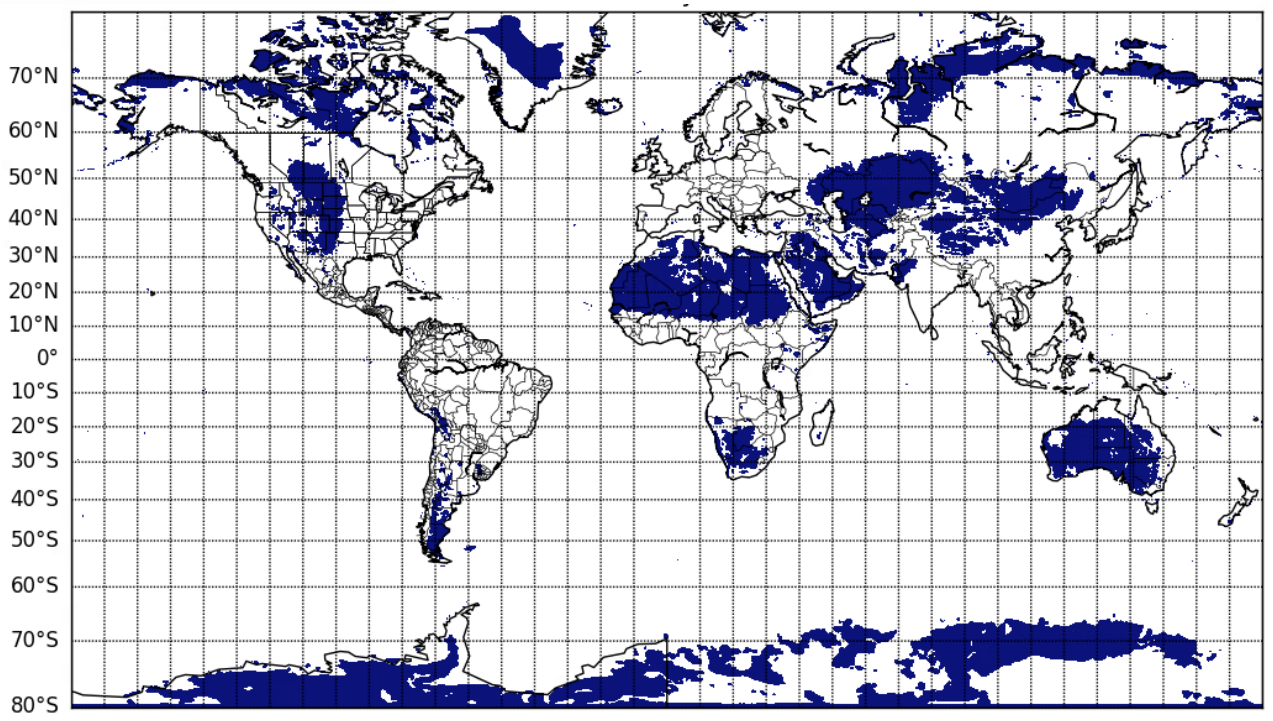
Integration into IFS-COMPO

The use of desert dust emissions has been tested into IFS-COMPO forecast only experiments with the following specifics :

- Cycle 50R1 + new dust emission scheme from CAMAERA WP5, adapted from SILAM + new dry deposition (proposed for 51R1)
- 2017 and second half of 2025 tested

Three experiments are shown:

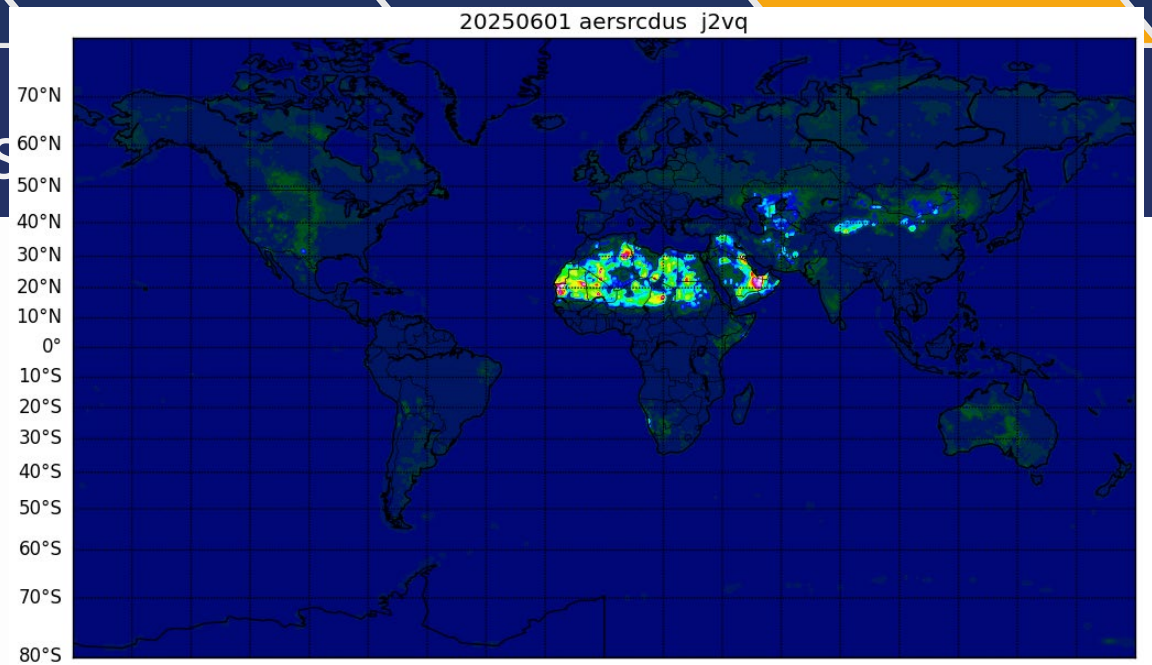
- Izww/j1ks : reference experiment using the physical scheme
- J2vq : experiment using the full 15 predictors neural network for dust emissions
- J53r : experiment using a similar configuration but only in the blue area – outside, desert dust emissions are always null.

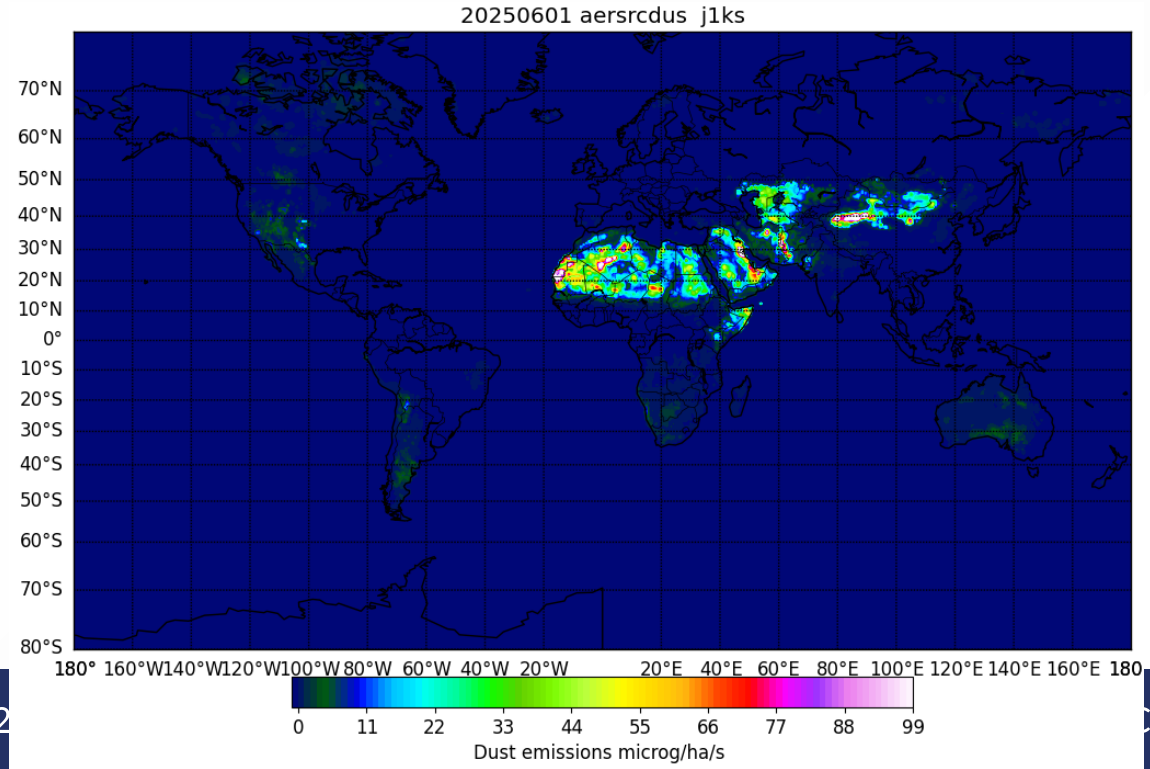


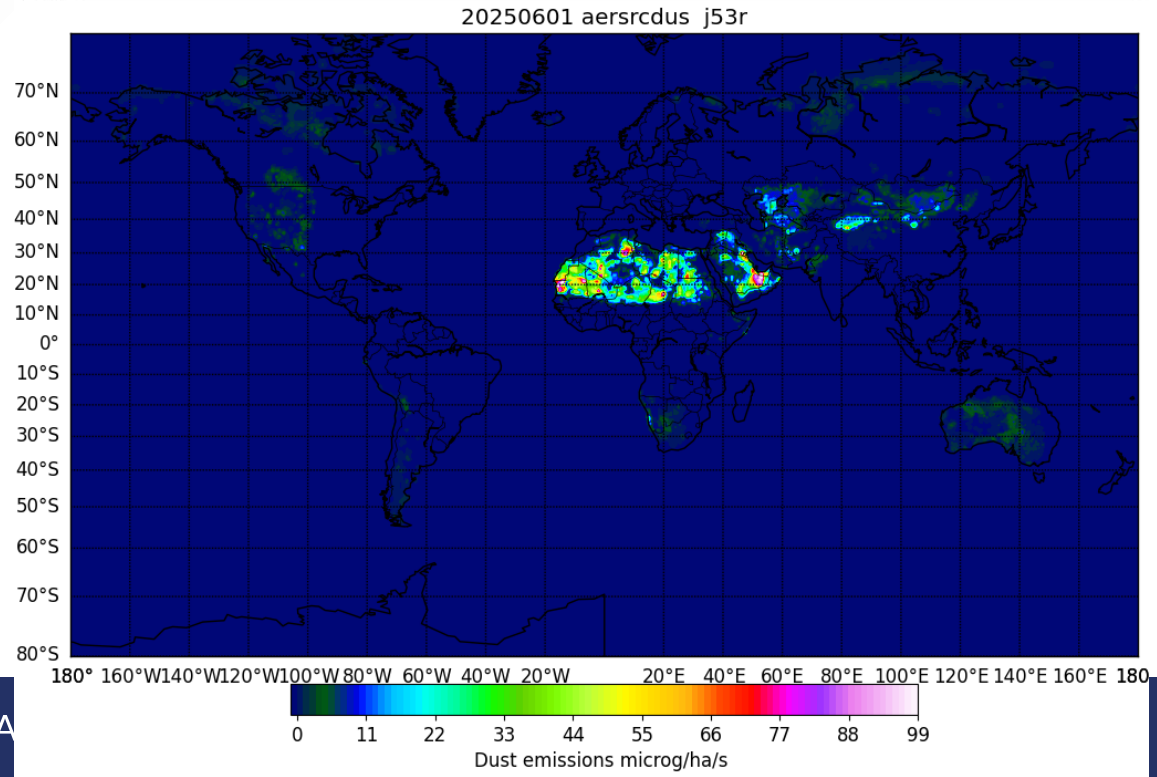
Simulated desert dust emis

Simulated dust emissions averaged in June 2025:

- General features are quite similar
- High latitude dust sources visible over Canada
- NN predicts lower emissions over Taklimakan
- For j2vq (15 predictors, no mask), large areas with low but non null emissions



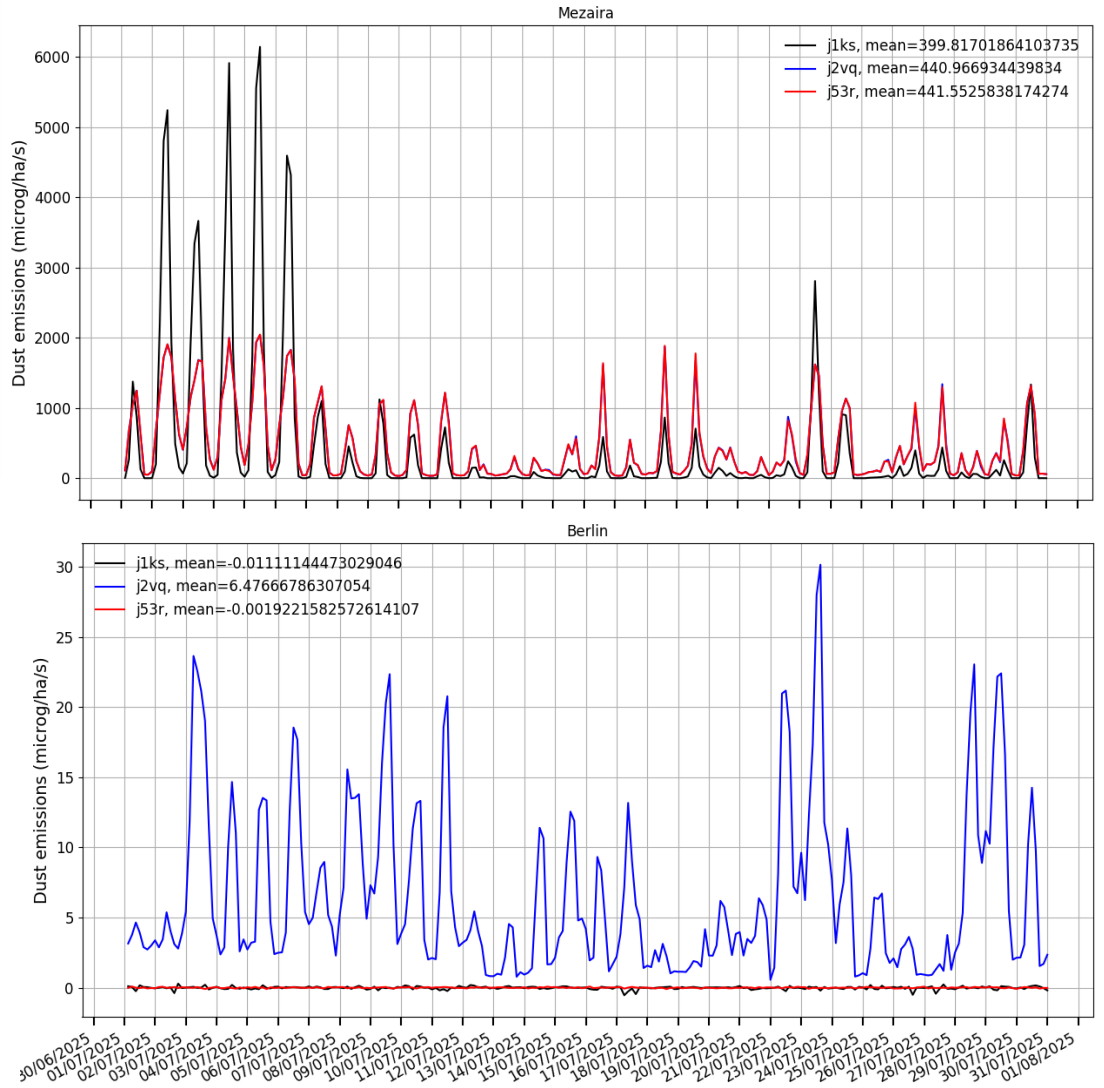




Simulated desert dust emissions in IFS-COMPO

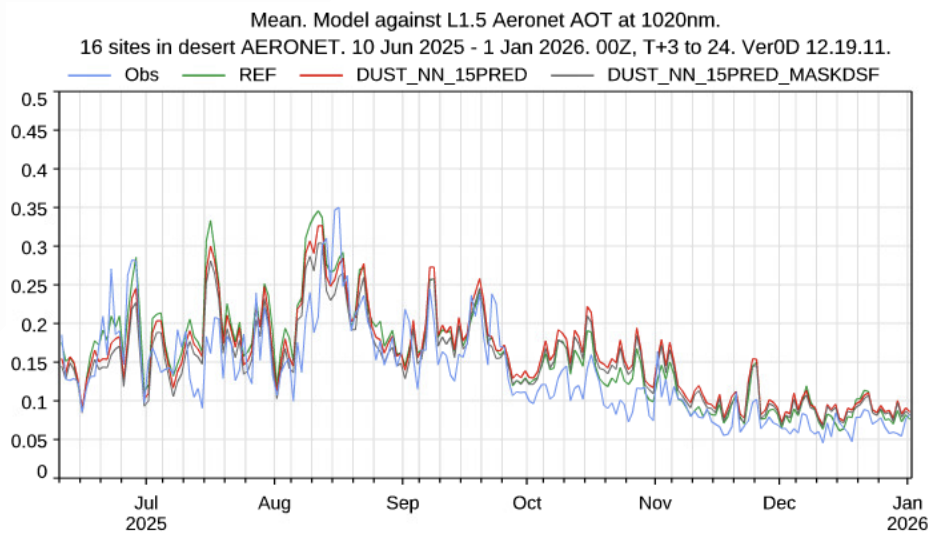
Timeseries of simulated dust emissions over Mezaira (UAE – dusty region) and Berlin (Germany, not dusty!) show:

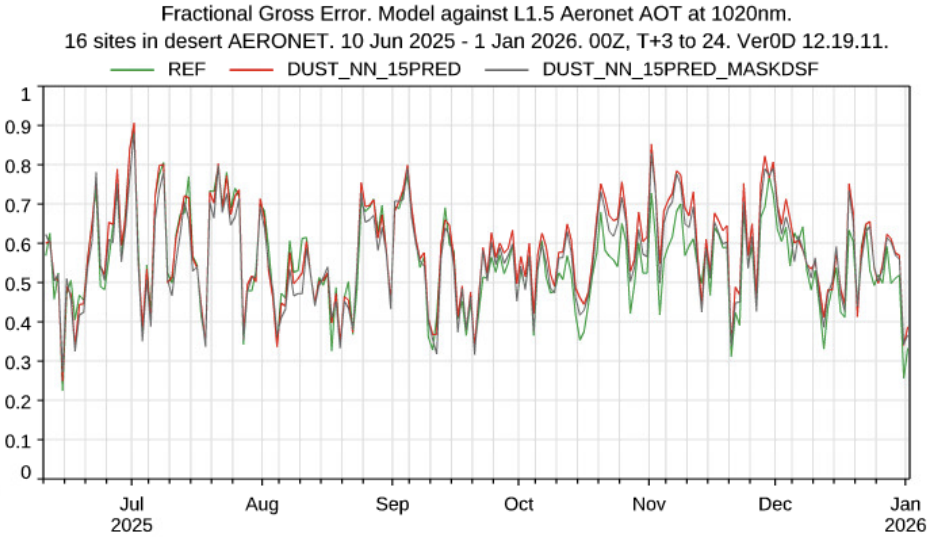
- NN without mask predicts low but still significant dust emissions over Berlin, effectively masked out with the use of a DSF based mask
- The occurrence of dust emissions in Mezaira is forecasted in quite a similar way
- The diurnal cycle shows a lower amplitude with NN : no null values at night, lower maxima in daytime



Impact on IFS-COMPO simulated AOD

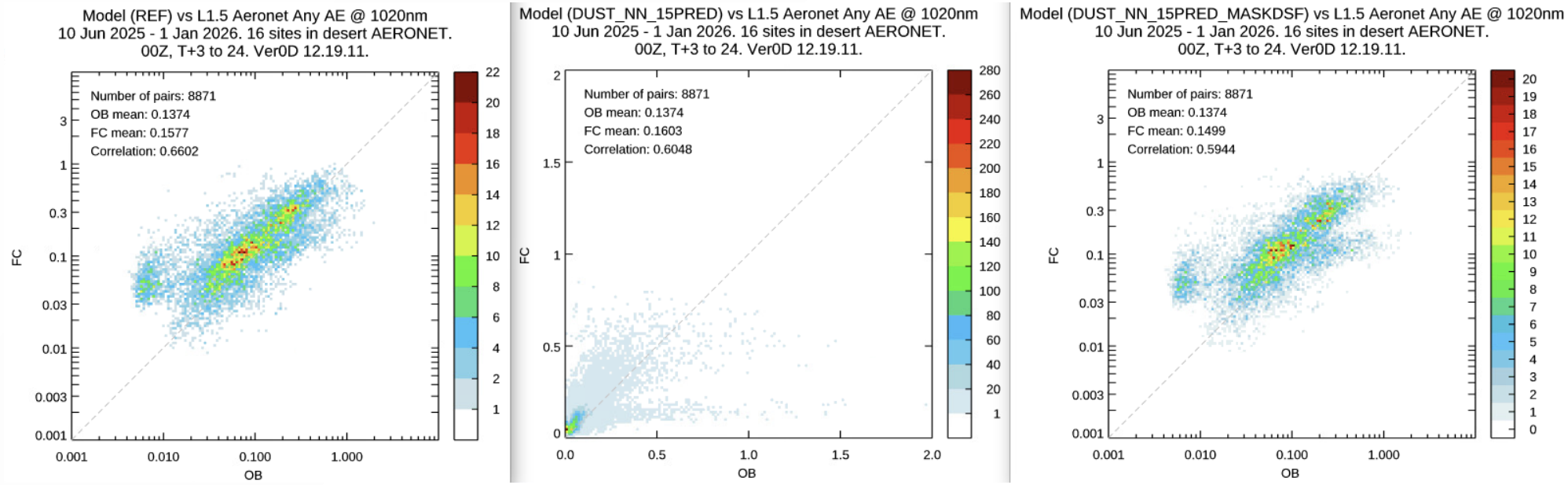
- A comparison of simulated AOD at 1020nm over a selection of « dusty » AERONET stations show comparable results
- No strong signal on error – sometimes better, sometimes worse





Impact on IFS-COMPO simulated AOD

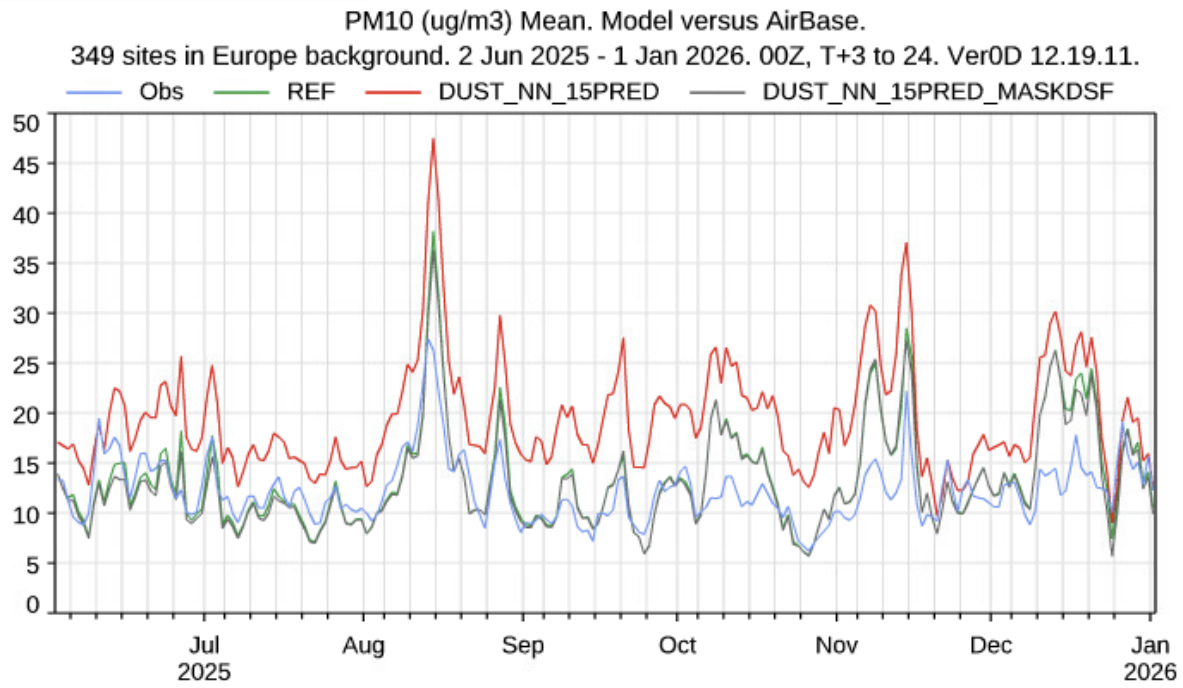
- A comparison of simulated AOD at 1020nm over a selection of « dusty » AERONET stations show comparable results
- No strong signal on error – sometimes better, sometimes worse
- Correlation is clearly degraded



Impact on IFS-COMPO simulated PM10

A comparison of simulated vs observed PM10 over background rural stations in Europe show:

- A clear degradation from the small and persistent emissions with the NN
- Problem easily removed by the masking
- Skill scores very close



Conclusion

Where we are ?

- Feed-forward neural network with 15 predictors for desert dust emission
- Optimization of the hyper-parameters according to the R^2 achieved
- Draft of Feature analysis (Permutation importance & Integrated Gradients)
- Integrate model inside IFS-COMPO via INFERO library and online evaluation of it

Planned improvement ideas :

- Rebalance the distribution of dust emission values in the dataset to encourage the model to focus on high emissions.
- Train a UNet model with 3x3 or 5x5 windows to identify if spatial information is useful. If yes, maybe add standard deviation of some predictors as input for the model.
- Tests with more elaborate physical constraints on dust emissions in IFS-COMPO

Training Description [Training loop]

Basically, the *training loop* alternates between **training** epochs (forward pass → loss computation → backpropagation → optimizer step) and **validation** epochs (evaluation on the validation set without gradient updates).

However, I added the following *callbacks* functions :

- **Early stopping** callback: monitors validation loss with a patience of 8 epochs
  - Training halts when validation loss stops improving
  - Best checkpoint is automatically saved and restored after training
- **Logging**: all training runs are monitored and metrics are sent to an local MLFlow server for tracking
- Model exported to **ONNX** for IFS-COMPO deployment

Features analysis [Permutation]

Goal: Understand which input features drive the model's predictions.

Permutation Importance — a model-agnostic method

1. Take the trained model and a sample of data
2. Shuffle one feature's values
3. Measure the drop in performance (MSE)
4. Repeat for each feature

Interpretation: A large performance drop means that the feature is important. Little to no drop means that the feature is redundant.

Feature	Importance	Magnitude
WINDGUST	1.97E-05	0.0008
USTAR	6.64E-06	0.0013
ALBEDO	3.76E-06	0.0005
SANDFRACTION	3.70E-06	0.0004
CLAYFRACTION	2.63E-06	0.0004
BARESOIL	2.44E-06	0.0003
z0_ASCAT	2.33E-06	0.0006
LAIL_iqsw	2.28E-06	0.0003
WINDSPEED	2.19E-06	0.0008
SOILWETNESS	1.45E-06	0.0002
OROSTDEV	1.03E-06	0.0004
Ut	9.84E-07	0.0004
ROUGHNESS	7.09E-07	0.0002
LAIH	4.87E-07	0.0004
SNOWDEPTH	4.61E-10	1.73E-07

Features analysis [Gradient]

Integrated Gradients — a gradient-based attribution method:

1. Start from a baseline (e.g., all zeros)
2. Compute gradients along a path from baseline to actual input
3. Integrate the gradients to get attribution scores per feature

Interpretation: Higher absolute attribution means that the feature has a stronger influence on the model's output for that prediction.

	Feature	Variation	Integrated Gradients
1.	USTAR	-	0.0007
2.	WINDSPEED	-	0.0003
3.	WINDGUST	-	0.0002
4.	SANDFRACTION	invariant	0.0002
5.	BARESOIL	invariant	0.0002
6.	ALBEDO	annual	0.0001
7.	CLAYFRACTION	invariant	0.0001
8.	Ut	invariant	9.84E-05
9.	ROUGHNESS	annual	6.62E-05
10.	OROSTDEV	invariant	6.26E-05
11.	z0_ASCAT	invariant	4.46E-05
12.	LAIH	annual	3.66E-05
13.	LAIL_iqsw	annual	3.20E-05
14.	SOILWETNESS	-	1.08E-05
15.	SNOWDEPTH	-	3.56E-08