

interTwin

D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

Status: Under EC Review
Dissemination Level: public



Funded by the
European Union


Disclaimer: Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them

Abstract

Key Words digital twin, environment, climate change, impacts, tools

This deliverable describes the release of Digital Twin (DT) Applications that support the climate change use cases detailing their implementation and validation report. It details the capabilities, characteristics, and describes the functional specifications of the DT applications and their integration into the DTE architecture. Finally, it provides the final validation of the developed and integrated DTs.



Document Description			
D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports			
Work Package 4			
Document type	Deliverable		
Document status	Under EC Review	Version	1
Dissemination Level	Public		
Copyright Status	 <p>This material by Parties of the interTwin Consortium is licensed under a Creative Commons Attribution 4.0 International License.</p>		
Lead Partner	CERFACS		
Document link	https://documents.egi.eu/document/3943		
DOI	https://zenodo.org/records/17106485		
Author(s)	<ul style="list-style-type: none"> • Christian Pagé (CERFACS) • Björn Backeberg (Deltares) • Willem Tromp (Deltares) • Donatello Elia (CMCC) • Marco De Carlo (CMCC) • Shahbaz Alvi (CMCC) • Martin Schobben (TU Wien) • Bernhard Raml (TU Wien) • Sandro Fiore (UNITN) • Iacopo Ferrario (EURAC) • Alice Crespi (EURAC) • Suriyah Dhinakaran (EURAC) 		
Reviewers	<ul style="list-style-type: none"> • Raul Bardaji (EGI Foundation) • Andrea Manzi (EGI Foundation) 		
Moderated by:	<ul style="list-style-type: none"> • Andrea Cristofori (EGI) • Andrea Anzanello (EGI) 		
Approved by	<ul style="list-style-type: none"> • Andrea Cristofori (EGI) on behalf of the TCB 		

Revision History			
Version	Date	Description	Contributors
V0.1	09/07/2025	Template creation from the 1st version of the DTs capabilities	Andrea Cristofori (EGI)
V0.2	5/08/2025	Version ready for internal review	All Authors
V0.3	26/08/2025	Internal review	Raul Bardaji (EGI Foundation), Andrea Manzi (EGI Foundation)
V0.4	31/08/2025	TCB review	Andrea Cristofori (EGI)
V1.0		Final	

Terminology / Acronyms	
Term/Acronym	Definition
Abbreviation	Full Form / Description
AI	Artificial Intelligence
AWI	Alfred Wegener Institut
CERRA	Copernicus regional reanalysis for Europe
CDS	Climate data store
CMIP6	Coupled Model Intercomparison Project Phase 6
CNN	Convolutional Neural Network
CVAE	Convolutional Variational Autoencoder
DL	Deep Learning
DNN	Deep Neural Network
DT	Digital Twin
DTE	Digital Twin Engine
ECMWF	European Centre for Medium-Range Weather Forecasts
EO	Earth Observations
ERA5	Fifth generation ECMWF reanalysis for the global climate and weather



ET	EvapoTranspiration
FAR	False Alarm Rate
FCCI	Fire Climate Change Initiative
GCN	Graph Convolutional Network
hPa	Hectopascal
IBTrACS	International Best Track Archive for Climate Stewardship
LAI	Leaf Area Index
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
ML	Machine Learning
MODIS	Moderate Resolution Imaging Spectroradiometer
MSE	Mean Squared Error
NetCDF	Network Common Data Form
NN	Neural Network
PLIA	Projected Local Incidence Angle
POD	Probability of Detection
Q	streamflow
RCP	Representative Concentration Pathways (set of scenarios)
SAR	Synthetic-Aperture Radar
SEAS5	Seasonal Ensemble Prediction System 5
SFINCS	Super-Fast Inundation of CoastS
SSIM	Structural Similarity Index Measure
SSM	Surface Soil Moisture
SSP	Shared Socioeconomic Pathways (set of scenarios)
STAC	SpatioTemporal Asset Catalogues
TC	Tropical Cyclone

D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

VAE	Variational Auto Encoder
Wflow	hydrological modelling framework
WP6	Work Package 6
xtclim	Generic Climate Extreme characterization and detection AI-based tool

Terminology / Acronyms: <https://confluence.egi.eu/display/EGIG>



Table of Contents

1	Introduction	11
1.1	Aim of this deliverable	11
1.2	For whom is this document	11
1.3	Structure of the document	11
2	DT Applications.....	12
2.1	Generic Detection of Climate Extremes.....	12
2.2	Wildfire danger prediction in response to climate change	13
2.3	Tropical storms change in response to climate change	15
2.4	Oceanic mesoscale eddies analysis	19
2.5	Flood early warning in coastal and inland regions	20
2.6	Alpine droughts early warning.....	22
2.7	Flood climate impact in coastal and inland regions	27
3	DT Applications Final release and Validation	30
3.1	Generic Detection of Climate Extremes.....	30
3.1.1	DT Application Integrations	30
3.1.2	Scope and limitations.....	31
3.1.3	Preconditions	34
3.1.4	Validation and Results	34
3.2	Wildfire danger prediction in response to climate change	35
3.2.1	DTs Application Integrations.....	35
3.2.2	Scope and limitations.....	38
3.2.3	Preconditions	41
3.2.4	Validation and Results	42
3.3	Tropical storms change in response to climate change	43
3.3.1	DTs Application Integrations.....	43
3.3.2	Scope and limitations.....	46
3.3.3	Preconditions	47
3.3.4	Validation and Results	47
3.4	Eddies detection	48
3.4.1	DTs Application Integrations.....	49
3.4.2	Scope and limitations.....	51
3.4.3	Preconditions	51
3.4.4	Validation and Results	51
3.5	Flood early warning in coastal and inland regions	52
3.5.1	DTs Application Integrations.....	52
3.5.2	Scope and limitations.....	54
3.5.3	Preconditions	55
3.5.4	Validation and Results	55



3.6	Alpine droughts early warning.....	56
3.6.1	DTs Application Integrations.....	56
3.6.2	Scope and limitations.....	61
3.6.3	Validation and Results	62
3.7	Flood climate impact in coastal and inland regions	62
3.7.1	DTs Application Integrations.....	62
3.7.2	Data requirements.....	64
3.7.3	Scope and limitations.....	65
3.7.4	Preconditions.....	66
3.7.5	Validation and Results	66
4	Conclusions.....	67
5	References	69

List of Figures

Figure 1	Results from the application of a trained ML model on CMIP6 data for the scenario SSP126, model CMCC-ESM2.....	15
Figure 2	Inference of ML model on TCs occurrence in 2014 for the basin of interest.	17
Figure 3	Four example patches were taken from the test set and fed to the trained model.	18
Figure 4	Topography and mesh of a FESOM2 setup with study region (black box)	19
Figure 5	Eddies detected in the Mediterranean from Sea Level Anomaly (SLA) with py-eddy tracking. Image courtesy of AWI	19
Figure 6	- a) Base configuration of the SFINCS model for the Darss peninsula built using local data. b) Example flood map produced by the SFINCS model from figure a)	21
Figure 7	Flood map for the Darss peninsula Germany on 21th of October 2023 using the TU Wien flood mapping algorithm	22
Figure 8	DT Application region and river basins in red	23
Figure 9	DT hydrological workflow linked components diagram	24
Figure 10	Simulated Evapotranspiration and SSM time series at three locations over the Alpine region	26
Figure 11	SFINCS model configuration for the Humber estuary, UK as shown in the notebook showing the topography, rivers, and discharge time series in a) and the coastal waterlevel timeseries and floodmap output in b)	28
Figure 12	DT impact visualization showing the aggregated direct damage calculated by Delft-FIAT in a) and optimal routes in the (un)disrupted road networks calculated by RA2CE in b)	29
Figure 13	Overview of the workflow for DT on extreme events.....	31
Figure 14	Template for the main configuration of the input data to the DT	32
Figure 15	Template for the configuration of Training and Model configurations and parameters.....	33
Figure 16	Template for the description of the DT workflow with parameters for each step	33
Figure 17	Annual Mean Losses for the 2m Daily Maximum Temperature	34
Figure 18	Number of days above threshold anomaly detection per year for the 2m Daily Maximum Temperature. Those anomalies represent climate extremes (very hot days)	35
Figure 19	Overview of the wildfires DT application workflow	37
Figure 20	Interface to choose the CMIP6 models and scenarios along with the years to include in the analysis.....	40
Figure 21	Interface to choose the aggregation of the 8-day burned area forecast on the seasonal, yearly, and decadal time-scale.	41



Figure 22 Results from different aggregation schemes for the 8-day burned area predictions applied on the CMIP6 model CMCC-ESM2 for the scenario SSP126.....	41
Figure 23 Average difference in terms of percentage of burned area per grid point on the test set between the values predicted by the ML model and the ground truth from the SeasFire cube.....	42
Figure 24 Overview of the Tropical Cyclones DT application	45
Figure 25 Interface to choose the CMIP6 models along with the years to include in the analysis.....	47
Figure 26 Results from the TC tracking applied on a subset of the CMIP6 model CMCC-CM2-VHR4.....	47
Figure 27 IBTrACS (in red) and detected TC trajectory (in blue) for the Keoni tropical cyclone	48
Figure 28 Eddies DT unified training and inference workflow	49
Figure 29 EddiesML notebook integrating the AI provenance tracking capability. As it can be seen the integration with yProv4ML calls (cell 3 and 28) is lightweight and does not significantly impact on the user's code.....	50
Figure 30 Traceability of the entire ML training process, documented via yProv4ML integration with a focus on a subset of the provenance graph, highlighting some of the metrics captured by the library at runtime	51
Figure 31 Validation phase pyEddyTracker vs Eddies DT inference	52
Figure 32 High-level workflow diagram for the flood early warning DT.....	54
Figure 33 SFINCS validation by comparing waterlevels observed at the green triangles in Figure 6b) (dashed lines) with the SFINCS output (solid lines).....	56
Figure 34 Agreement between Global Flood monitor and SFINCS for a small section of the Darss peninsula without (a) and with (b) dike breaches in the SFINCS model.	56
Figure 35 High-level diagram of the DT workflow's application components.....	59
Figure 36 High level diagram that shows how DT's application components are run.....	59
Figure 37 The image shows the volumetric water content (VWC, i.e. SSM) for three initialization dates, February, April and June, averaged over the Po basin.	62
Figure 38 Architecture diagram highlighting integration and interaction between the Jupyter Notebook (the DT interface), the CWL workflows and interTwin's RUCIO-based datalake.....	64

List of Tables

Table 1 Initial set of variables considered for training	30
Table 2 SeasFire Cube and corresponding CMIP6 data variables identified to carry out the wildfires prediction case study	35
Table 3 CMIP6 data from ScenarioMIP project made available on the interTwin Data Lake	36
Table 4 Input parameters to the ML model	43
Table 5 CMIP6 data from HighResMIP project made available on the interTwin Data Lake	44
Table 6 Input Parameters from Open Source Datasets.....	57
Table 7 Processed input-output of wflow_sbm	58
Table 8 Dataset to perform parameter learning	58
Table 9 Datasets used by the DT.....	64



Executive summary

This document is deliverable 4.7 of the interTwin project, part of work package 4. It is a report collectively written by the partners of tasks 4.5, 4.6 and 4.7, who have been directly involved in the design and implementation of Digital Twin (DT) Applications for the environmental domain (climate projections & extreme events). In the deliverable, for each DT, there is a description of integration with the DTE, the data used, followed by discussions on the scope and limitations of each implementation. A discussion on the validation of each DT is also described. The integration with the Infrastructure Components provided by WP5 and the Core Components provided by WP6, along with schematic high-level workflows are shown.

Overall, each DT is specifically designed and linked to address climate change impacts and provide valuable insights for assessing climate risks, identifying early warning signals, and implementing mitigation measures. It provides invaluable tools enabling users to explore several possible specific impacts of climate change.

1 Introduction

1.1 Aim of this deliverable

The overall objective of deliverable 4.7 is to provide information about the capabilities of DT Applications related to climate change and impact decision support tools. Those DTs are from the environmental domain (T4.5, T4.6, T4.7). A **DT Application** is a user-facing implementation of a DT. DT applications are the consumers of the capabilities offered by the interTwin DTE, thus introducing use case-specific requirements.

1.2 For whom is this document

The deliverable 4.7 can be useful for both developers and end users as described below:

For **developers**: the deliverable provides them with an opportunity to be informed on how the integration of DTs has been planned and implemented. This integration is also linked to the different capabilities and features of each specific DT. It gives insights on how those DTs fit into the overall interTwin architecture by using specific core components.

For **end users**: the specified deliverable provides information on the capabilities of each specific DT in the environmental domain. It provides insights to the users on what can be achieved by using those DTs: what they can be used for, and eventually what are the parameters and configurations that the end users can set for tailoring those DTs to their specific needs. By establishing a common framework for communication, researchers and stakeholders will be able to exchange information, validate models, and collaboratively address climate change impacts and suggest mitigation measures.

1.3 Structure of the document

The structure of this deliverable is as follows. **Section 2** describes the capabilities of each DT application. **Section 3** explains how the integration with the DTE has been performed and the results obtained. The reader should refer to D4.5 [R12] for a description of the architecture.

2 DT Applications

2.1 Generic Detection of Climate Extremes

This DT is aimed at the detection and characterization of climate extremes to assess their impacts and provide useful information for the decision-making process. The DT enables users to quantify and evaluate the changes of climate extremes, such as changes in the frequency of occurrence, as well as their spatial extent, duration, and intensity. The DT is using the maximum daily temperature variable to calculate extreme temperatures compared to the normal values, according to a specific season or monthly period, depending on users' choices. The method is an anomaly detection method that is generic and can be applied to any other atmospheric climate variables such as precipitation and wind speed.

The end users are able to select specific greenhouse gas scenarios to explore different impacts according to the evolution of those emissions that are driven by national and international policies. Users also have the capability to select a region of interest to focus on specific areas and better evaluate regional and local impacts. It is also possible to select a specific time period and a season (or a monthly period), as well as several datasets coming from different types of future climate simulations, such as global circulation coupled models, regional climate models, and any other relevant datasets.

To evaluate the changes, the end users define a time period of reference to compare with. Typically, a period of 25 to 30 years is used as a climate reference, in order to better evaluate the needed adaptation compared to a specific state of the climate.

This DT is based on Artificial Intelligence (AI) techniques. It is a novel technique for this kind of analysis. It unlocks the possibility for users to better quantify the uncertainties associated with different sources such as greenhouse gas emission scenarios, specific climate models and also inherent climate variability. This is possible thanks to the high performance of Deep Learning (DL) methods in processing large datasets. Typically, using analytical methods, users would only use a small subset of all available simulations, leading to partial uncertainty assessments resulting in incomplete impact assessments. This could lead to less-than-optimal adaptation and mitigation decisions.

In more detail, the DL method that is used consists of a Convolutional Variational Autoencoder (CVAE). Autoencoder and decoder are a pair of unsupervised trained Neural Networks (NNs) where autoencoders are trained to compress the data and decoders are trained to decompress the compressed data with a minimal loss. This can be used for anomaly detection, hence also for climate extremes. When input data is compressed, the main features of the data are kept, and there is some loss since, after compression, data is stored in a lower-dimensional space (called latent space) before being decompressed. Variational Autoencoders (VAE) model the latent space as a probability distribution. CVAEs use Convolutional Neural Networks (CNN) for both the encoder and decoder parts.

In this implementation, CVAE input data is a subset of CMIP6 [\[R9\]](#) data: a time sequence of 32 X 32 square images of a daily average of a specific atmospheric surface variable



over a specific spatial subset region. Training is done by season, for each specific climate model separately, using a time period long enough to have sufficient samplings (typically 50 years or more) and in which the greenhouse gas emissions have a weak tendency (e.g., 1850-1950). The CVAE model can then be applied to any time period of interest to the user, using any CMIP6 simulation data of this specific global climate model. For each daily image reconstructed by the CVAE, there is a loss, corresponding to the error in the reconstructed image. This loss value is used for anomaly detection (climate extremes are anomalies).

2.2 Wildfire danger prediction in response to climate change

Several studies show that climate change will affect both the frequency and severity of wildfires. Modelling fire regimes therefore is important in assessing future potential impacts on the social and economic aspects of society. Machine Learning (ML) algorithms have emerged recently as effective alternatives for the prediction of wildfire occurrences compared to the traditional approaches (e.g., dynamic global vegetation models). The advantage ML models enjoy over traditional models is due to the fact that data-driven models can learn complex interactions allowing them to provide accurate predictions and unravel potential relationships between variables.

The DT related to the wildfires application focuses on the generation maps of areas burned due to forest fires on a global scale. These maps are generated using ML models which learn the spatial distribution of historical burned areas on a global scale from their correlation with certain predictors. While we have experimented with several ML architectures, the principal architecture adopted in this work has been the *UNET++*¹. The networks are trained to learn the non-linear spatial relationship between multiple environmental variables and the percentage of the pixel area burned. The variables used as covariates for percentage burned area include weather and vegetation conditions.

The data for training and validation is derived from SeasFire Cube², a scientific datacube containing 21 years of data (2001-2021) with an 8-day temporal resolution and 0.25° spatial resolution, designed to forecast seasonal fires around the world. The fire predictors are upscaled to 100 km resolution (~1°). The fire predictors from the SeaFire cube are stacked of dimension of $H \times W \times C$, where H is the height and it is equal to 720, W is the width and it is equal to 1440, C is the number of climatic variables. In the final model implemented, 5 fire predictors have been used namely, Leaf Area Index, Land-Sea mask, Relative humidity, Temperature at 2 metres - Min, Total precipitation), The target is Burned Areas from the ESA Fire Climate Change Initiative (FCCI)³.

The trained ML model can then be applied to past data from the SeasFire Cube or on future climate projection data. Such data has been ingested from the ScenarioMIP project of the Coupled Model Intercomparison Project, phase 6 (CMIP6) initiative, which collects

¹ <https://arxiv.org/pdf/1807.10165>

² <https://zenodo.org/records/8055879>

³ <https://climate.esa.int/en/projects/fire/>



Earth system model simulations according to alternative plausible emissions and land use future scenarios [R11]. The esgf_rucio component from WP7 has been configured to consume the data from the ESGF data nodes⁴ and upload it on the interTwin Data Lake (WP5).

The DT application addresses two types of end-users: scientists with technical skills that want to train a new ML model and less technically skilled users that want just to run the DT application. Details of the training process, the architecture, data preparation and target users are provided in D4.5 [R12].

Keeping in view the target end-users from the first group: researchers and developers, the training pipeline is made highly configurable. The hyperparameters of the training, and other related parameters (like using a different architecture) are easy to configure. For instance, different use cases might be interested in different performance metrics. Such metrics can consist of (i) well-known metrics from a Python library (torchmetric⁵, scikit-learn⁶, etc.) or (ii) “customized” metrics coded in Python to be used within the ML4Fires thematic module (WP7) and tracked during the model training and validation.

Training of the ML model can be performed through a simple bash script. MLflow library⁷ is used to track the training progress and log the models, their hyperparameters, and other artifacts. The model logging helps track different ML experiments and analyze the results. Moreover, provenance of the model is also tracked during the training workflows. These two features are enabled thanks to the integration of the itwinai (WP6) logger capabilities.

Inference is an important part of the validation of the ML model. Moreover the inference plays an important role in the work of both policy makers and researchers. The interface for performing inference for a logged model is also provided in the form of a Jupyter notebook⁸. Multiple notebooks are available in particular:

- One notebook allows the end-user to create inference maps for a period in the SeaFire dataset. This allows the user to perform inference and validation of the trained model with respect to the observation data available in the SeaFire dataset. The notebook is configured with easy-to-use widgets and tools to post-process the inference, enabling them to draw various conclusions from the inference maps of burned areas.
- The second notebook allows the second class of users to run the model on the CMIP6 dataset. The Jupyter Notebook allows the researcher to produce inferences using the climate projection data from the ScenarioMIP CMIP6 dataset as the input to the ML model. The notebook includes a flexible way to choose the CMIP6 model, scenario and range of years for which the forecast is needed. The post-processing of the inference (how to aggregate the 8-day burned area over the yearly or decadal scale) can also be performed. Such processes can be executed on multiple

⁴ <https://esgf.github.io/nodes.html>

⁵ <https://github.com/Lightning-AI/torchmetrics>

⁶ <https://github.com/scikit-learn/scikit-learn>

⁷ <https://mlflow.org>

⁸ https://github.com/CMCC-Foundation/ML4Fires/tree/main/digital_twin_notebooks



files using Ophidia⁹ workflows¹⁰. An example of a map produced by the notebook is shown in **Figure 1**.

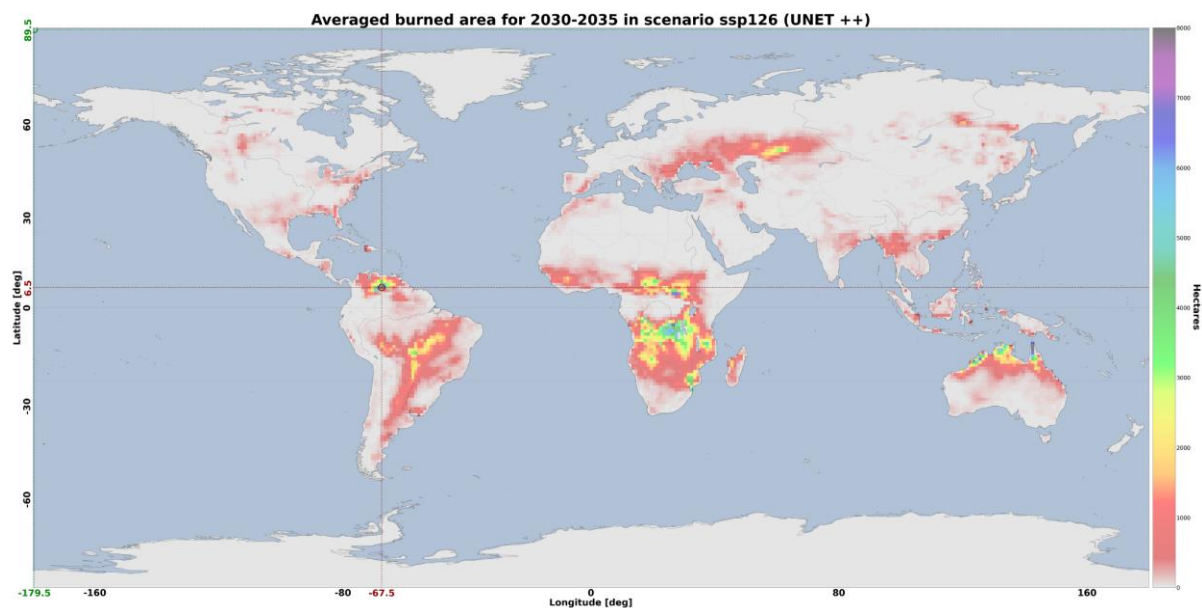


Figure 1 Results from the application of a trained ML model on CMIP6 data for the scenario SSP126, model CMCC-ESM2. The 8-day burned area prediction is aggregated over the selected 5 years time range

2.3 Tropical storms change in response to climate change

Tropical Cyclones (TCs) rank among the most powerful and destructive natural hazards, often leading to severe disruptions and extensive losses across the globe. Their pronounced sensitivity to climate variability and their considerable socio-economic consequences make the precise tracking and forecasting of TCs a persistent and complex challenge in both meteorology and climate research. This DT focuses on the detection and tracking of TCs and consists of classifying the absence or presence of a cyclone in gridded climate fields and in a specific time instant and, if present, localising its centre (or “eye”) in terms of latitude/longitude geographical coordinates.

ML models are used to learn the mapping between climatic fields significant to the cyclogenesis and the positions and trajectories that storms follow during their lifetime in historical records. Trained models are exploited to detect the occurrence of storms in future projection scenarios to indicate, across both space and time, the occurrence of such phenomena.

Since the variable representing cyclogenesis is composed of a set of 2-dimensional (2D) data, this can be interpreted as a 2D image, where each pixel corresponds to a cell of the

⁹ <https://ophidia.cmcc.it/>

¹⁰ <https://github.com/CMCC-Foundation/ML4Fires/tree/main/workflows>

lat-lon grid. Starting from this consideration, two different approaches have been explored for the detection of TCs: CNNs and Graph Convolutional Networks (GCNs).

Concerning the geographical domain, the joint North Pacific formation basin was targeted because it is considered the one with the highest number of TCs occurrences yearly. Six input climatic variables (i.e., mean sea level pressure, 10m wind gust since previous post-processing, instantaneous 10m wind gust, relative vorticity at 850 mb, and temperature at 300 and 500 mb) have been initially gathered from the ERA5 reanalysis datasets¹¹ for the region of interest. Data has been downloaded from the Copernicus Climate Change Service¹². These variables were used as predictors of TC presence, stacked together, and treated as separate channels of a multi-channel image. The temporal extent considered is 1980–2019, using ERA5 data with a temporal extent of 6 hours. To improve model efficiency and, crucially, to ensure that each sample presented to the NN contains at most one tropical cyclone, a patch-based segmentation is applied as a pre-processing step. Each reanalysis map, composed of 280×880 grid points, is preliminarily divided into 7×22 non-overlapping patches of 40×40 .

In the CNN approach, the ML model identified for the DT is a Visual Geometry Group (VGG)-like CNN [R1, R2]. The final version of the model has been trained on just two drivers (i.e., mean sea level pressure and relative vorticity at 850 mb) allowing to reduce the volume of data to be fed to the network. The experimental workflow is structured in two main stages:

- I. A combination of two DL models (i.e., a classification model and a localization model) is employed to identify the position of TC centers;
- II. A multi-object tracking algorithm is applied to associate TC detections over time, thereby reconstructing complete TC trajectories (Figure 2).

In the first stage, the classification model assigns a probability of TC presence to each spatial patch, using a detection threshold of 0.5. For patches classified as positive, the localization model estimates the coordinates of the TC eye within the patch. In the second stage, the tracking algorithm links these spatial detections across consecutive time steps to form consistent spatiotemporal cyclone tracks. Previous implementations have explored the use of an ensemble of ML models for improving the ML models' results (see previous deliverable D4.5 [R12]).

¹¹ <https://www.ecmwf.int/en/forecasts/dataset/ecmwf-reanalysis-v5>

¹² C3S: <https://climate.copernicus.eu>

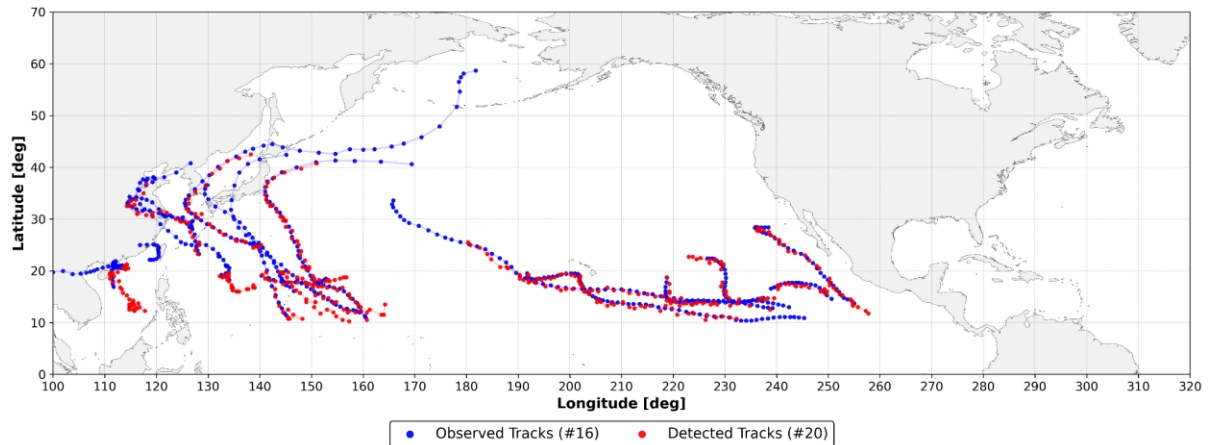


Figure 2 Inference of ML model on TCs occurrence in 2014 for the basin of interest. Red markers represent the positions identified by the ML model that was fed with two climatic variable patches for each time step of the TC trajectory. Blue markers denote the actual positions of the TC throughout its trajectory extracted from IBTrACS.

In the second case a GCN-based approach has been investigated. The model can output a (row, col) coordinate in the range $[0-39]$, $[0-39]$ if a TC is detected (i.e., positive patch). Instead, negative labels (e.g., $(-1, -1)$) are predicted when no cyclone is found within the patch (i.e., negative patch). The IBTrACS¹³ dataset has been used to map each positive patch with the corresponding TC's historical occurrence. While the stacking of the atmospheric variables and the division into patches is identical to the CNNs case, the data at this stage is organised in grids, not in graphs. An additional step is hence needed to make the patches readable by the GCN, and it consists of retrieving the adjacency information from these matrices, linking together the pixels as if they were nodes in a graph. Along with this information, the features are permuted from their current dimension, $C \times H \times W$, to $L \times C$, where L is the list of graph nodes with dimension $W \times H$, rather than a matrix.

Together with the coordinates (lat, lon) information of the cyclones, the dataset also provides probability density maps that are still derived from IBTrACS and range from 1.0 (the pixel with the cyclone) to 0.0 (the furthest locations from the eye of the turbulence). To maintain the compatibility of the pipeline with a previous version of the code, during CNN supervised training we use the matrix coordinates, whereas in GCN learning we use the probability density maps instead. Therefore, the GCN will be trained to find a non-linear mapping between 3D input graphs and the position of the corresponding maximum probability value in output.

¹³ <https://www.ncei.noaa.gov/products/international-best-track-archive>

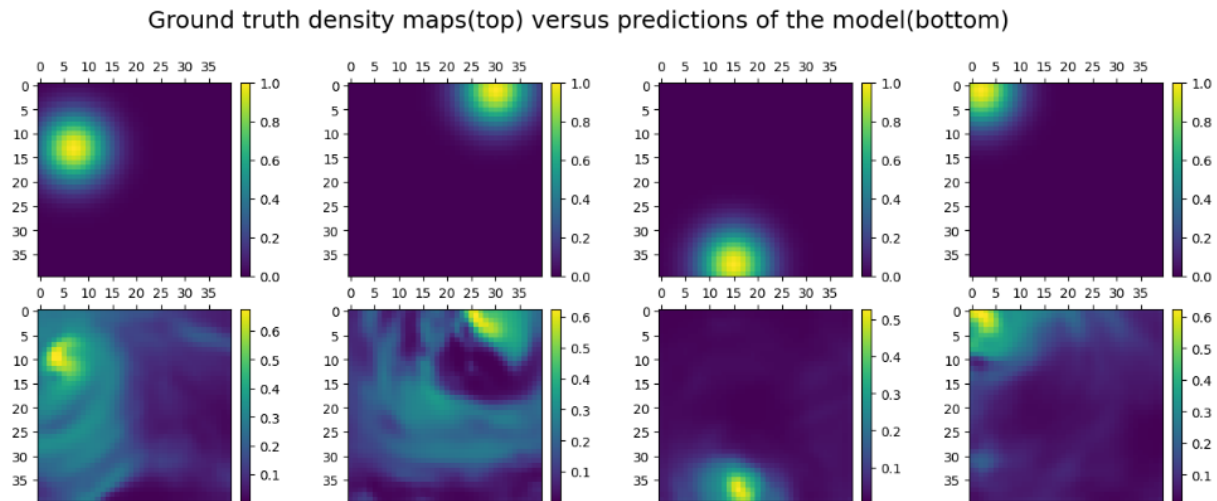


Figure 3 Four example patches were taken from the test set and fed to the trained model. The top row shows the actual positions of the cyclones, while the bottom contains the probability estimates of the positions produced by the GCN model. The colour bars represent the confidence of the prediction from 0 to 1

The ML models can be trained through a simple bash script. The configuration in ML TC detection modules (WP7) is used to identify the model setup (i.e., CNN or GCN). itwinai (WP6) logging capabilities are used to track the training progress on MLflow, jointly with the main model hyperparameters and the resulting artifacts. Provenance is also logged using yProv4ML (WP6).

End-users can run the trained ML models through notebooks¹⁴ to perform analysis of TCs on current and future climate data and visualise the results through maps and charts as well as download/save them as NetCDF files. Multiple notebooks are available in particular:

- The first notebook allows users to create inference maps of TCs on ERA5 data and compare them with IBTrACS observations. This allows the user to visually validate the model results. Widgets are provided to select the time period of interest among the test data. Additional plots are provided to further validate the data according to different metrics (e.g., duration of tracks, percentage of tracks detected, etc.);
- The second notebook allows the end-users to run the model on the CMIP6 dataset. Before running the inference in the notebook, a preprocessing pipeline needs to be executed. This has been implemented as a Python script and as an Ophidia workflow for processing multiple datasets together. The notebook allows the researcher to produce inferences using the climate projection data from the HighResMIP CMIP6 dataset¹⁵ as the input to the ML models. Similarly to the wildfires use case, it includes a flexible way to choose the CMIP6 model and the range of years for which the detection is needed.

¹⁴ <https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection/tree/main/notebook>

¹⁵ https://highresmip.org/experiments/experiment_cmip6

2.4 Oceanic mesoscale eddies analysis

The Eddies DT addresses oceanic mesoscale eddies analysis. Ocean mesoscale eddies are the “weather” of the ocean, with typical horizontal scales of less than 100 km and timescales on the order of a month.

This DT can be considered an example of “exploitation” use case, as it applies interTwin technologies to a DT provided by an external institution during the project lifetime; in particular the ML pipeline comes from the Alfred Wegener Institut (AWI) and it is based on unstructured grid data from the FESOM2 model (Figure 4 shows an example of topology and mesh of a FESOM2 setup).

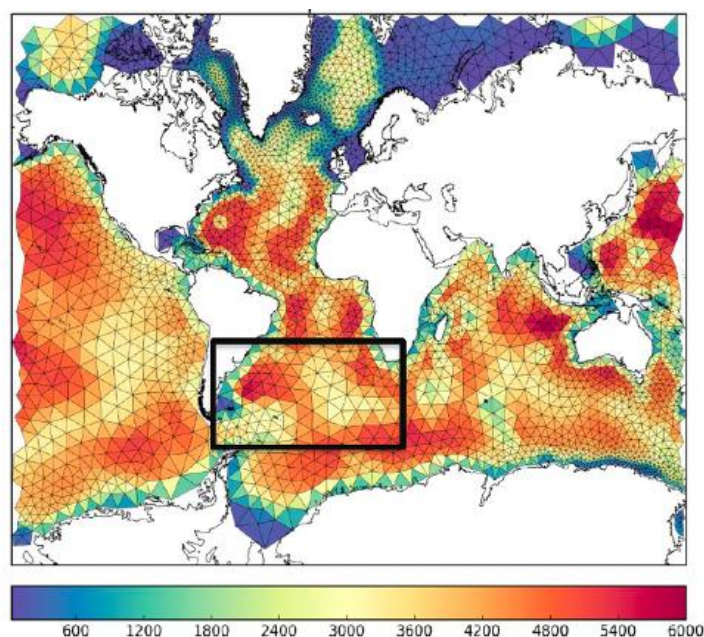


Figure 4 Topography and mesh of a FESOM2 setup with study region (black box). Image courtesy of AWI

The goal is to provide Jupyter Notebooks for expert users (environmental scientists) in order to select the input data, spatial domain, temporal target, pre-trained ML model and then run the DT inference step. By properly configuring the input parameters in the Jupyter Notebook the users can then run the Eddies DT to perform the ML task.

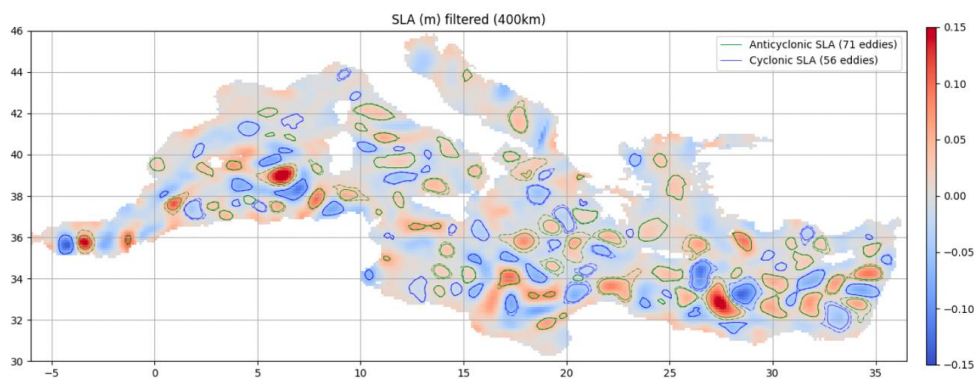


Figure 5 Eddies detected in the Mediterranean from Sea Level Anomaly (SLA) with py-eddy tracking. Image courtesy of AWI

One key aspect of the approach introduced by interTwin has been to move towards a data-driven paradigm. A ML algorithm learns from the ground truth generated by the py-eddies-tracking package, which can be considered the state of the art algorithm to detect eddies and it is based on the closed contours of SSH. The inference is then much faster than the data-intensive application of thepy-eddies-tracker. The two workflows are easily implemented by leveraging interTwin technologies. Extra features like provenance support further document the ML training process.

The DT reduces the complexity for the end users to perform eddies analysis. To this end, the thematic modules of interTwin facilitate the setup of the necessary models and workflows.

In the scope of these activities on eddy detections, a researcher from UNITN spent three weeks in Bremerhaven, Germany, at the Alfred Wegener Institut, working with the other researchers on the Eddies DT workflow.

2.5 Flood early warning in coastal and inland regions

This DT focuses on the post event analysis of flood events, integrating flood maps derived from satellite observations with those of a numerical inundation model. Identifying the mismatches between the two types of flood maps allows a user to fine-tune their numerical inundation model, for instance by fine-tuning the location of potential dike breaches.

End users will be able to specify a geographic region of interest and interTwin's core and thematic modules will enable setting up the flood inundation SFINCS (**Figure 6**), including necessary Earth Observation data processing pipelines to monitor and predict floods for the user-defined region of interest. The DT runs the models and processing pipelines to generate flood maps, provide indicators for the overlap between the two types of flood maps, and allows a user to adjust the inundation model with dike breach or overtop locations. Users will also have the capability to add their own local data to the model schematisations, thus enhancing model accuracy (e.g., Digital Elevation Models based on local LiDAR surveys).

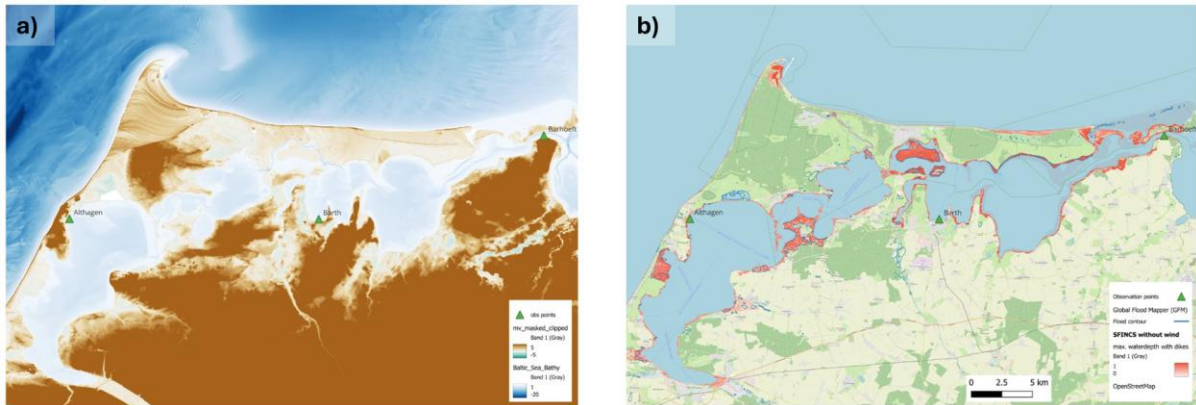


Figure 6 - a) Base configuration of the SFINCS model for the Darss peninsula built using local data. b) Example flood map produced by the SFINCS model from figure a)

The DT's flood maps are produced by SFINCS¹⁶ (Super-Fast Inundation of CoastS), a reduced-complexity model for super-fast dynamic modelling of compound flooding. SFINCS is forced by weather forecasts. Additionally, the DT combines the SFINCS flood maps with Sentinel-1-based flood maps generated by the Dask-flood-mapper, which implements the TU Wien flood mapping algorithm [R7]. This algorithm enables near real-time mapping of the flood extent through Bayesian inference from Sentinel-1 Synthetic-Aperture Radar (SAR) microwave backscattering. Globally applicable flood signatures are obtained through establishing predefined probability distributions of pixels under flood and non-flood conditions. These conditions are inferred on the generalised backscattering characteristics of water and land. The end user can run the dask-flood-mapper and derive flood maps for the targeted region (Figure 7). An example of the workflow for Sentinel-1-based flood mapping in Python syntax has been published on the interTwin GitHub¹⁷.

¹⁶ <https://www.intertwin.eu/article/thematic-module-sfincs>

¹⁷ <https://intertwin-eu.github.io/dask-flood-mapper/>

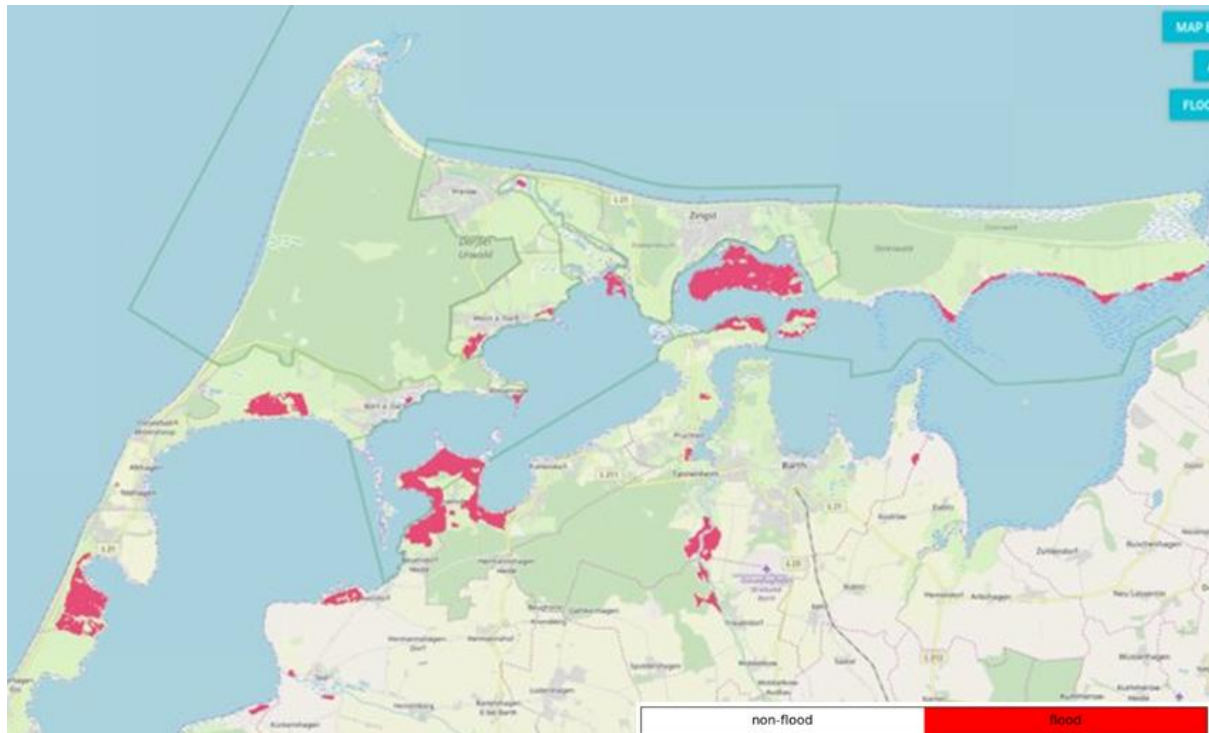


Figure 7 Flood map for the Darss peninsula Germany on 21th of October 2023 using the TU Wien flood mapping algorithm

The thematic modules of interTwin facilitate the setup of the necessary models and workflows anywhere on Earth. Workflow execution and offloading to remote compute infrastructure (WP5) is facilitated by the core modules (WP6).

2.6 Alpine droughts early warning

This DT aims at developing a drought early warning system for the Alpine region, providing seasonal forecasts of daily time series and maps of key hydrological variables such as Surface Soil Moisture (SSM, %), actual EvapoTranspiration (ET, mm) and streamflow (Q, m³/s). SSM and ET are produced at 1 km over the entire Alpine region whereas Q is produced at the outlets of the Alpine River basins (**Figure 8**).

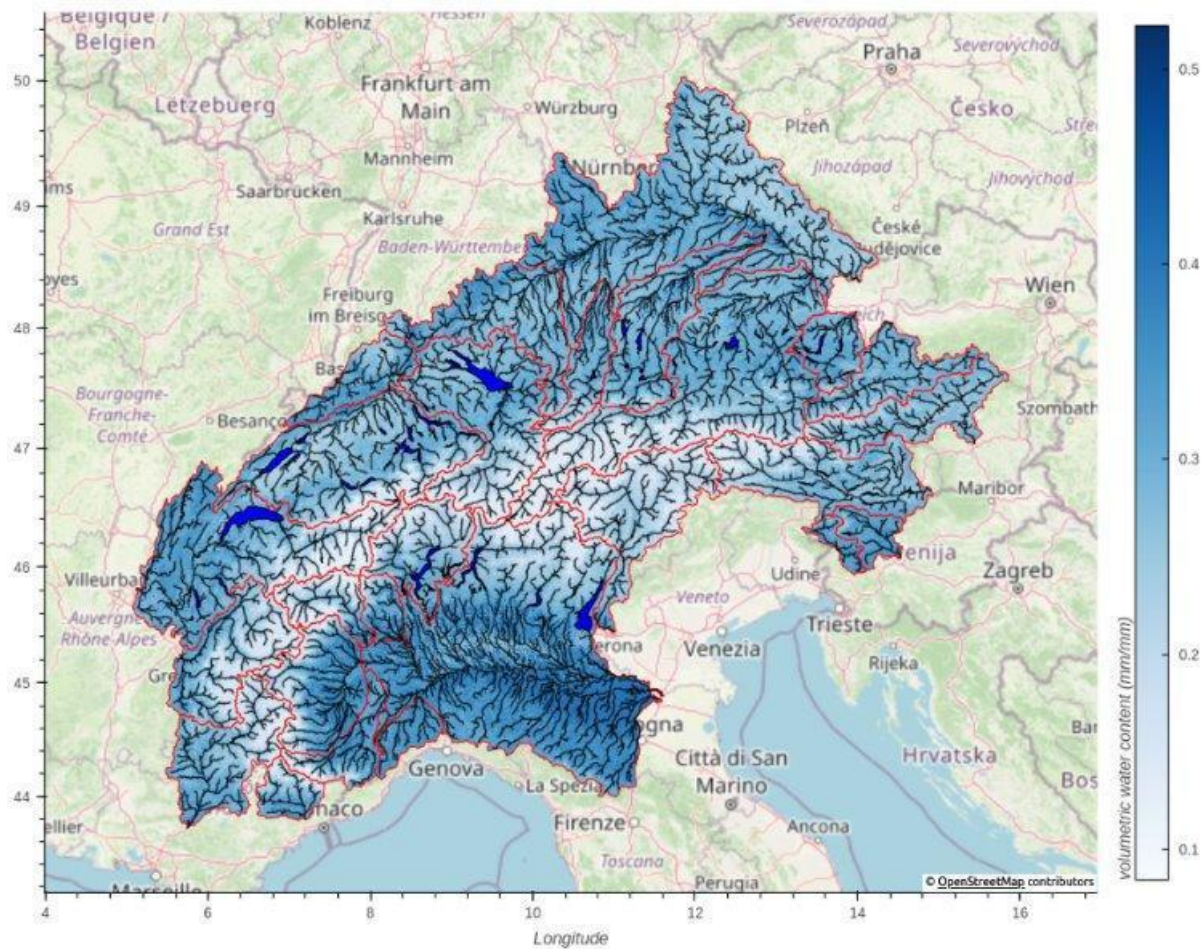


Figure 8 DT Application region and river basins in red

In summary, the DT receives meteorological variables from the ECMWF's SEAS5 seasonal forecast system [R3] and feeds them into a hydrological workflow to predict hydrological variables up to 6 months in the future. New seasonal forecasts are produced at every beginning of the month, enabling the monitoring of the onset, propagation and termination of hydrological droughts.

The users are able to set up and customise the DT's hydrological workflow through the openEO interface (WP6) and run experiments and visualise the outputs by means of Jupyter Notebooks.

The DT draws inspiration from the recent progress in hybrid modelling, where the relative strengths of data-driven algorithms and physical knowledge are combined and complemented. The injection of physical knowledge occurs in the initialisation of the data-driven model parameters, an approach that is known as physics-guided machine learning [R4]. The data-driven surrogate is trained to emulate the input-output mappings of a distributed hydrological model, wflow_sbm (WP7). The surrogate inputs are wflow_sbm's effective parameters and meteorological variables. Once the surrogate successfully emulates wflow_sbm, it can be further fine-tuned by means of observations



or, as it is currently done in the DT, it can replace wflow_sbm for performing extensive calibration of its parameters.

There are two properties of the physics-guided approach that are particularly advantageous: the computational performance and the flexible nature of DL architectures. The surrogate can run, in prediction, several orders of magnitudes faster than wflow_sbm, increasing the possibility of exploring larger regions of the parameter space to find optimal parameter sets. The DL models are also flexible as they can be adapted to different types and numbers of inputs, and they can be (relatively easily) composed and coupled with other NNs or differentiable models [R5].

The DT hydrological workflow consists of several logically linked components (Figure 9), which are to some degree customisable by the users. The workflow starts with the ingestion of SEAS5 and ends with the prediction of hydrological droughts.

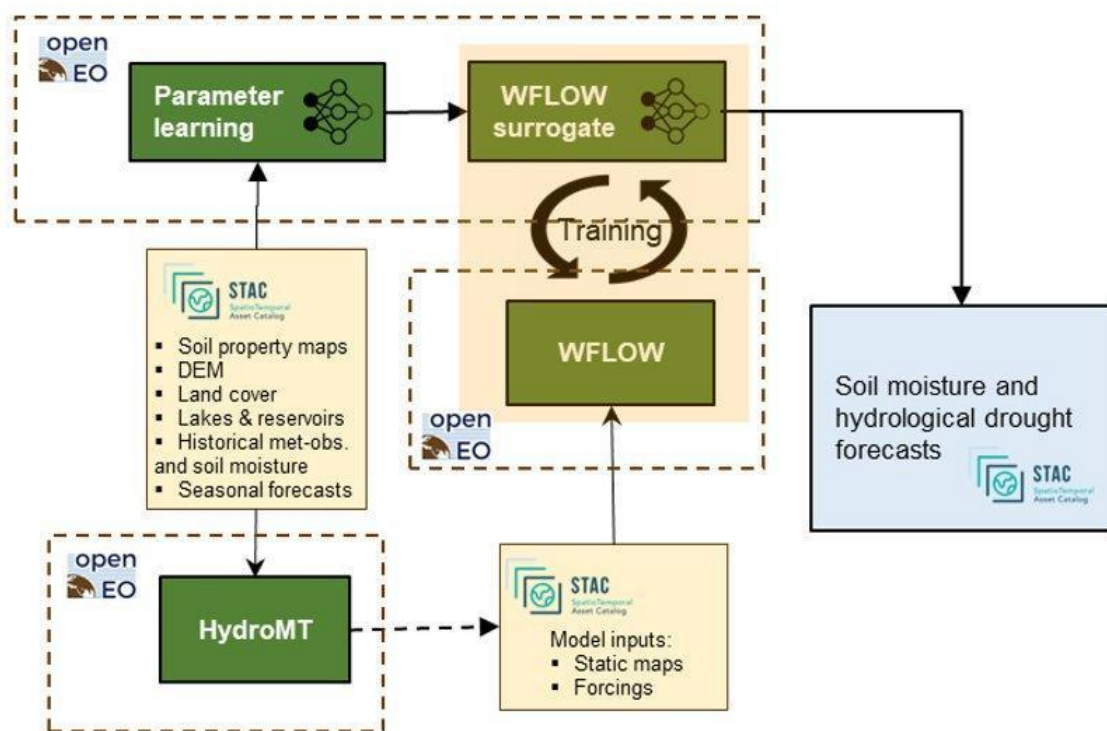


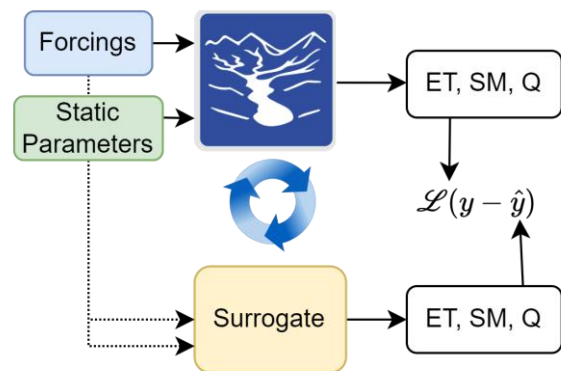
Figure 9 DT hydrological workflow linked components diagram

The *first component* of the workflow initially downscales meteorological variables from the SEAS5 forecast for better skills in the Alpine region using the 'downscaleML' python package (WP7). This package, stemming from WP7.4, currently includes routines for data preprocessing and for building statistical downscaling models for temperature and precipitation fields using a predefined dataset as a reference.

The *second component* is responsible for setting up and building the hydrological model, wflow_sbm, leveraging on the HydroMT¹⁸ application, a software for automating model building based on configuration files. The current release offers the option to select two precipitation inputs (E-OBS¹⁹ and CERRA-LAND²⁰), to change model resolution, and to filter input data temporally and spatially.

The *third component* deals with running wflow_sbm, which is a semi-distributed hydrological model developed by Deltares and recently rewritten from Python to the Julia language, to boost performance. One of the strengths of the model is that most of the effective parameters can be estimated by pedo-transfer functions using readily available global datasets. The effective parameters of the model are conceptual constructs that cannot be observed. Nonetheless they can be functionally derived from properties and attributes of the basin (i.e. topography, vegetation structure, etc.).

The *fourth component* is responsible for training the surrogate model. The surrogate is a Long Short-Term Memory (LSTM) NN which has been outperforming established physically based hydrological models in streamflow prediction tasks [R6]. As wflow_sbm spatial support is a grid of 1 km resolution, the LSTM model is trained over a representative subsample of its grid cells, to improve the training efficiency (maximise learned information and minimise training time). In every training loop, LSTM learns both SSM and ET at each grid cell (Figure 10) and Q at the outlet of the river basin. The loss function is a weighted sum of the individual losses of SSM, ET and Q.



¹⁸ <https://github.com/Deltares/hydromt>

¹⁹ <https://cds.climate.copernicus.eu/cdsapp#!/dataset/insitu-gridded-observations-europe>

²⁰ <https://cds.climate.copernicus.eu/cdsapp#!/dataset/reanalysis-cerra-land>

D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

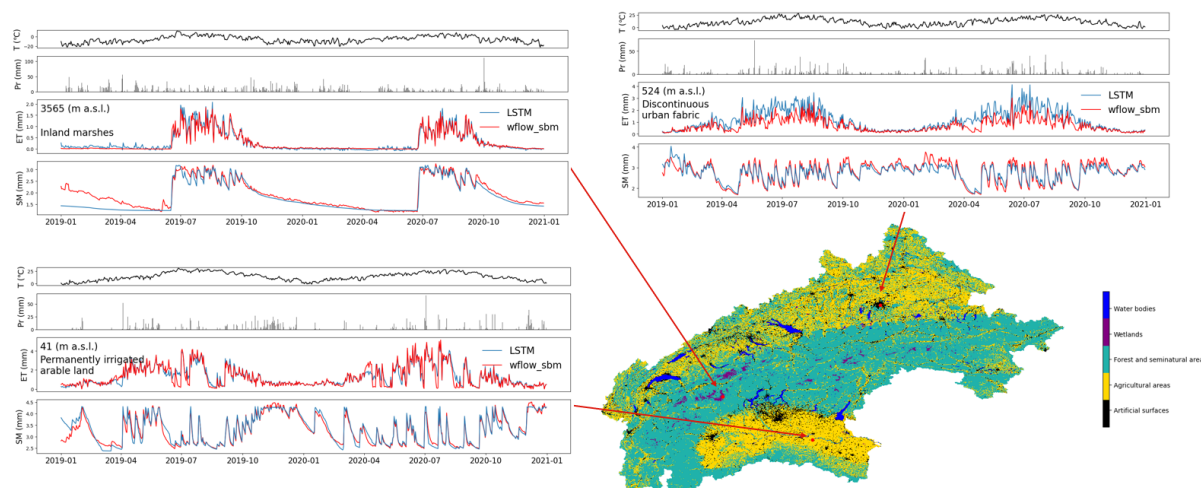
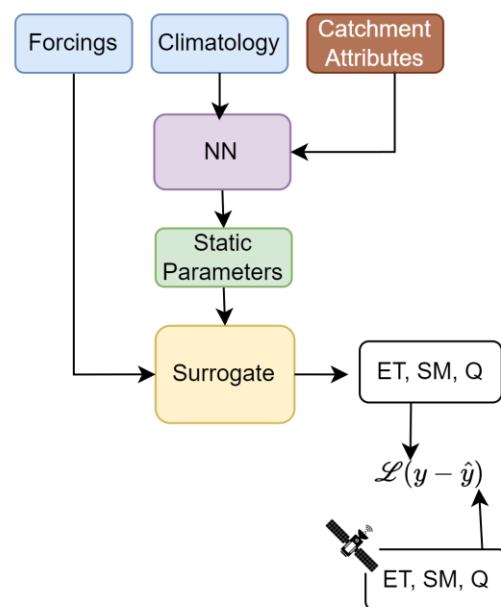
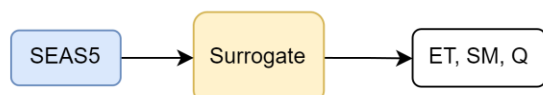


Figure 10 Simulated Evapotranspiration and SSM time series at three locations over the Alpine region

The *fifth component* deals with the parameter learning task. This task enables the calibration of the wflow_sbm parameters by training the surrogate model coupled with a NN that is responsible for learning the transfer function between the catchment attributes and the effective parameters. In this task the surrogate model weights are frozen and a CNN encoder is trained end-to-end to produce the optimal set of effective parameters. The CNN encoder is learning a transfer function from observations to effective model parameters. The loss function is based on the Mean Squared Error (MSE) between the simulated SM and the satellite-based SM retrievals (produced by TU Wien), and the simulated Q and the observed Q from the Alpine Drought Observatory (ADO)²¹ database.



Finally, the set of optimal parameters can be used with the surrogate or with wflow_sbm to predict the seasonal forecasts of SSM, ET and Q.



²¹ <https://ado.eurac.edu/>

2.7 Flood climate impact in coastal and inland regions

The DT for Flood Climate Impact in Coastal and Inland regions focus on simulating (compound²²) flood events by producing their flood hazard maps and assessing their impact on building, utilities, roads, and accessibility. Additionally, the DT enables end users to assess the efficacy of flood adaptation and mitigation measures for historic events and for future climate and socio-economic scenarios, such as sea level rise and population growth.

End users can define a geographic region of interest which will configure the necessary hazard and impact models and processing workflows using interTwin's core and thematic modules. Additionally, users can:

- Select weather events to be simulated
- Select future changes to scenarios including physical projections (such as sea level rise or land subsidence)
- Select socio-economic projections (such as population growth and economic growth),
- Select adaptation and mitigation measures to be implemented for a given scenario (e.g., flood walls, pumps, levees, culverts, buyouts, flood proofing, and/or raising properties).

Aside from using data provided through interTwin's Data Lake, users also have the capability to add their own local datasets (e.g., local DEM, building footprints, critical infrastructure) into the DT workflow, enhancing the models' accuracy.

The DT model chain comprises four different models. Firstly, the flood hazard maps are produced by SFINCS, a reduced-complexity hydrodynamic model calculating flood extents. The user describes an event to simulate which determines the meteorological data used as boundary conditions for SFINCS and for the second model Wflow²³, a hydrological model calculating river discharges for SFINCS to use. The flood maps produced by SFINCS are then ingested into the third and fourth models, Delft-FIAT²⁴ and RA2CE²⁵, which assess the direct damage to buildings and roads and the compounding impact on infrastructure networks respectively. The climate and/or socio-economic

²² Compound flooding refers to a situation where multiple flooding sources combine to exacerbate the overall flood impact. These sources can include:

- Fluvial Flooding: Flooding from rivers and streams due to heavy rainfall or snowmelt.
- Coastal Flooding: Flooding from storm surges, high tides, or sea-level rise affecting coastal areas.
- Pluvial Flooding: Flooding caused by intense rainfall overwhelming drainage systems, not necessarily linked to a body of water like a river or sea.

When these sources coincide, their combined effects can lead to more severe flooding than would occur from any single source alone.

²³ <https://www.intertwin.eu/article/thematic-module-wflow>

²⁴ <https://www.intertwin.eu/article/thematic-module-delft-fiat>

²⁵ <https://www.intertwin.eu/article/thematic-module-ra2ce>

D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

scenarios and flood adaptation measures described by the user are translated into updated boundary conditions for these models by the FloodAdapt backend.

The end user is provided with Jupyter Notebooks as an interface to the DT, which guides a user through the steps for configuring and running the DT. First, the user provides an area of interest based on which the models are configured (**Figure 11**). Next, the user describes the event to run by selecting meteorological data and providing a description of a climate and/or socio-economic scenario. The event will then be run by the model chain. Finally, the user can visualise and interact with the output data, either through the provided interactive visualisation notebook, examples of which are shown in **Figure 12**.

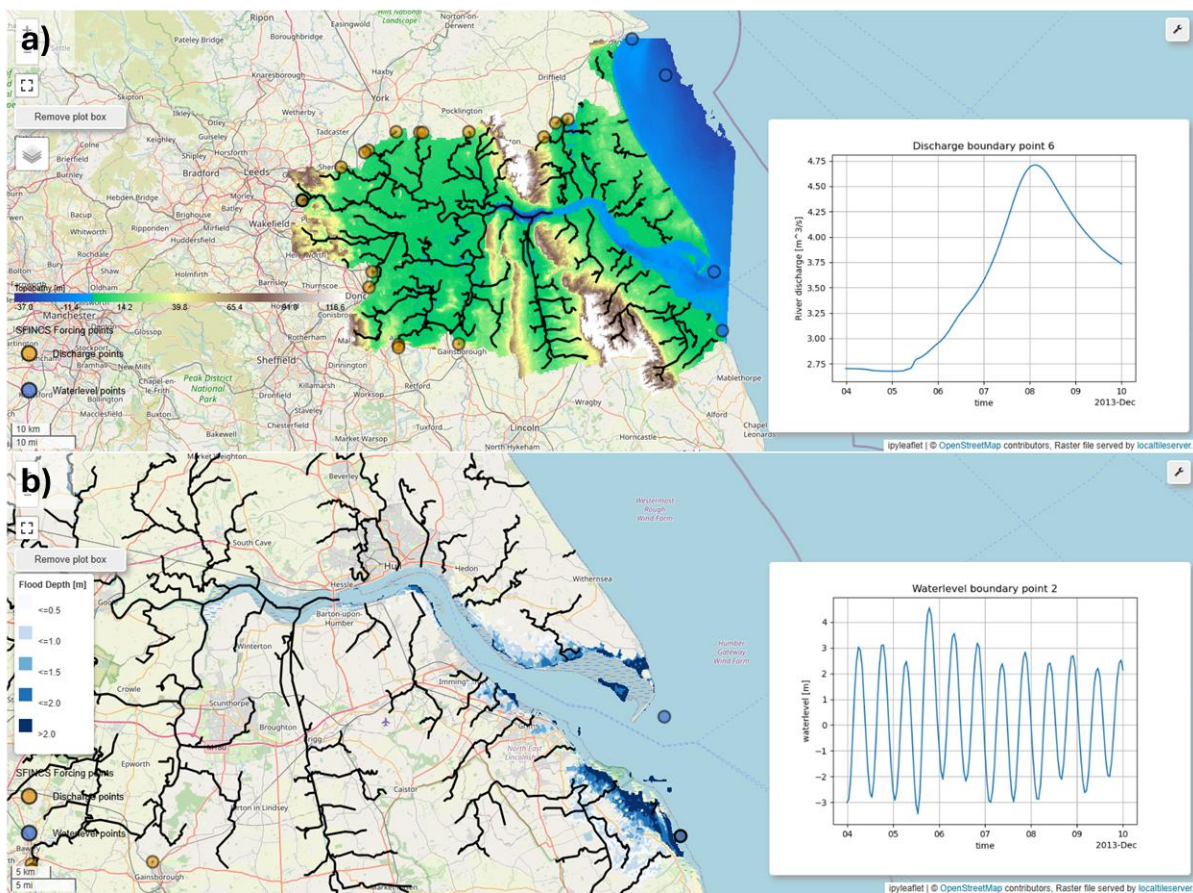


Figure 11 SFINCS model configuration for the Humber estuary, UK as shown in the notebook showing the topography, rivers, and discharge time series in a) and the coastal waterlevel timeseries and floodmap output in b)

D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

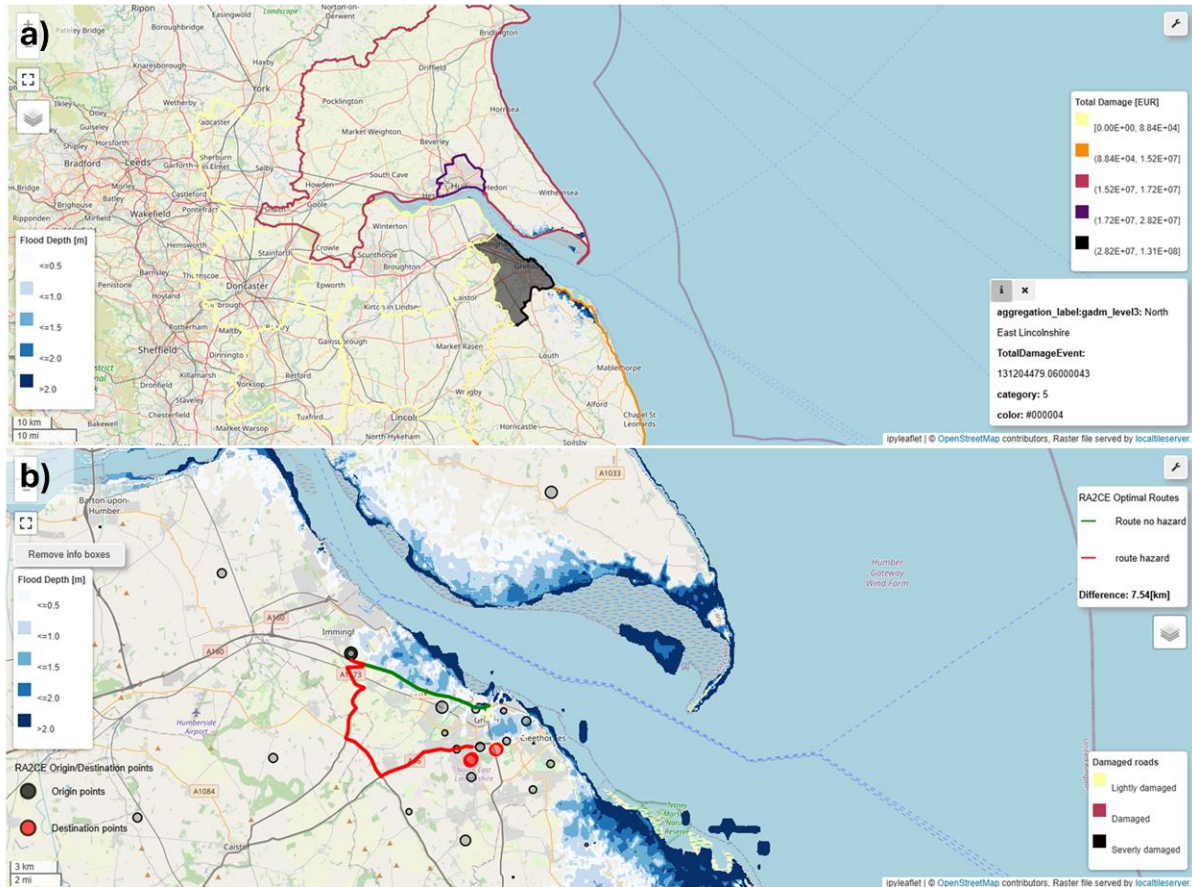


Figure 12 DT impact visualization showing the aggregated direct damage calculated by Delft-FIAT in a) and optimal routes in the (un)disrupted road networks calculated by RA2CE in b)

3 DT Applications Final release and Validation

3.1 Generic Detection of Climate Extremes

3.1.1 DT Application Integrations

Model requirements

The ML model uses as input data a subset of daily climate variables from the CMIP6 dataset. The implementation is for the maximum daily surface temperature, in order to address heatwaves and very hot days. This DT can be used with other temperature, precipitation and wind atmospheric variables, using the fact that the anomaly detection method is generic.

Table 1 Initial set of variables considered for training

Full name	CMIP6 variable name	Unit	Implemented
Daily Maximum Surface Temperature	tasmax	Kelvin	Yes
Daily Total Precipitation	prtot	kg/m ² /s	No, but supported
Daily Near Surface Wind Speed	uvas	m/s	No, but supported

Workflow

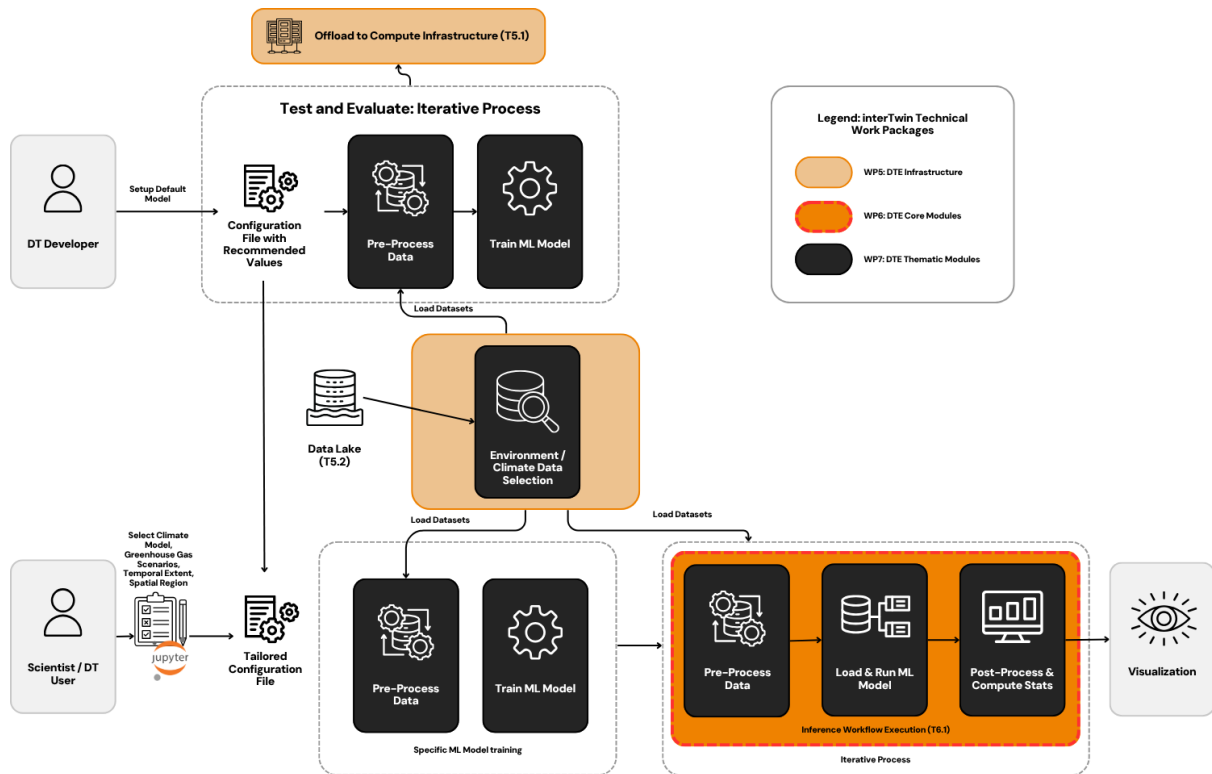


Figure 13 Overview of the workflow for DT on extreme events

The workflow about the Generic Detection of Climate Extremes DT exploits the following components from the project:

- WP7: The workflow depends on the modules available in the *xtclim* thematic module²⁶. The Jupyter Notebook implementation directly uses the functions made available by the library.
- WP6: The integration has been completed with the *itwinai* framework. *xtclim* is integrated as a plug-in to the DTE framework.
- WP5: Data for training and inference are accessed from a local demonstration dataset (CMCC-ESM2 GCM CMIP6 data: historical, ssp126, ssp245, ssp370, ssp585 for the period 1950-2100). The integration with the interTwin Data Lake is implemented as an alpha stage: initial feasibility tests have been successfully performed to evaluate the access to a sample set of CMIP6 data using the RUCIO and xarray Python modules. Implementation is not done due to the late availability of the data and technical issues with the cluster that delayed development and testing.

3.1.2 Scope and limitations

The current implementation of this DT targets climate extremes and impacts that can be calculated daily, and that are related to climate indices and percentile-based indicators. This DT can serve as a base to extend the current implementation to other climate extremes that involve a time dimension, such as droughts, or also to compound extreme

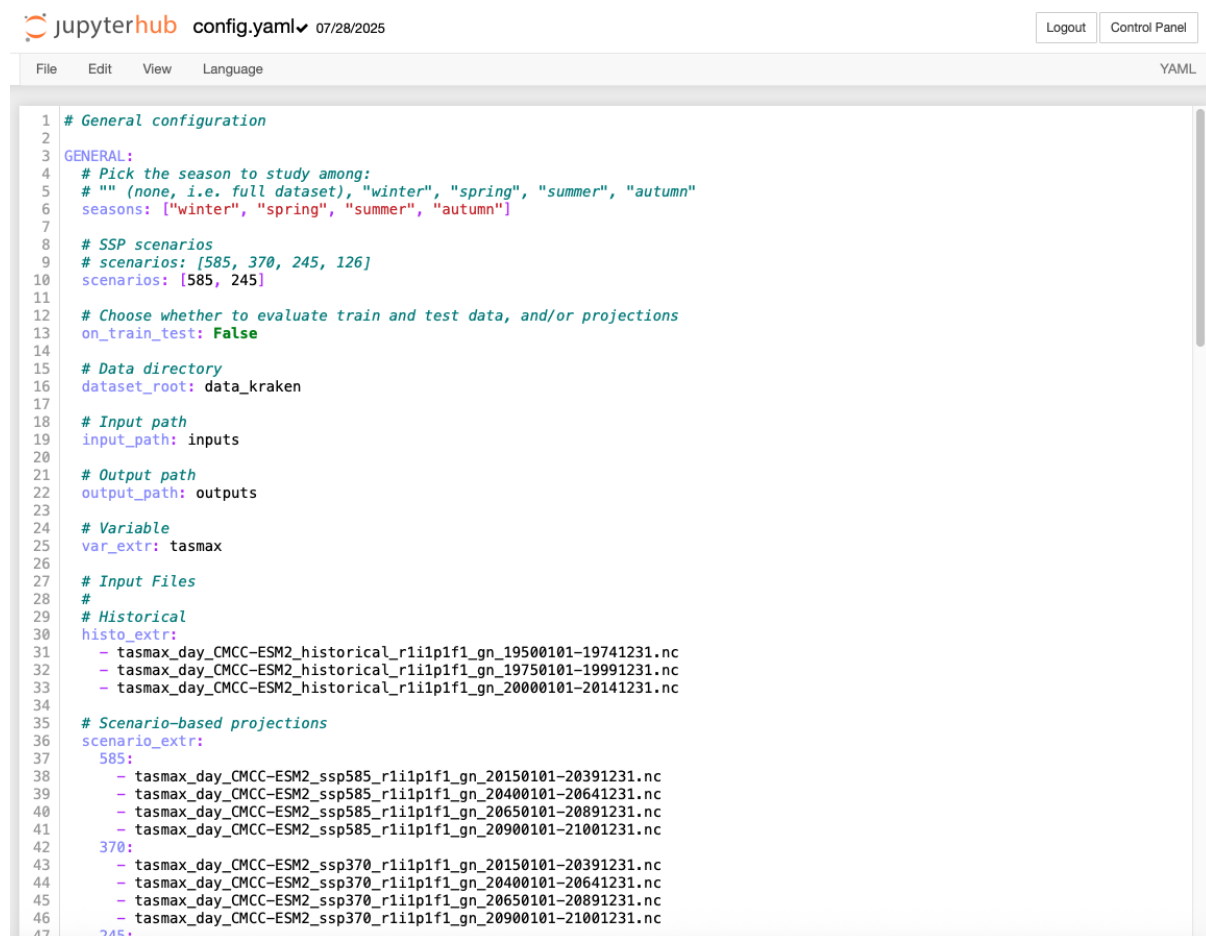
²⁶ <https://github.com/interTwin-eu/xtclim>

events.

Main limitations:

- Hyperparameters need to be optimised and adjusted for each atmospheric variable.
- Training must be conducted over a region with relatively homogeneous atmospheric variables and must be done season by season and separately for each climate model, using a long enough time period in which the climate change signal is relatively low and stable.
- Current configuration of the implementation is for a 32x32 input data array only. This is suitable to ensure that the statistics of the climate variable is sufficiently uniform in order for the DL method to detect anomalies.

The notebook is used for pre-processing the input data, performing the training of the ML model and to perform projection of the model using the CMIP6 dataset. Scripts are provided for running all or parts of the workflow. The configuration of the parameters (Figure 14 and Figure 15) and the workflow (Figure 16) is done through a self-described `config.yaml` file. Main user parameters are also described in the notebook and in the documentation README.md. A sample `config.yaml` file is provided to users to modify according to their input data and atmospheric variables they want to analyze.



```
1 # General configuration
2
3 GENERAL:
4 # Pick the season to study among:
5 # "" (none, i.e. full dataset), "winter", "spring", "summer", "autumn"
6 seasons: ["winter", "spring", "summer", "autumn"]
7
8 # SSP scenarios
9 # scenarios: [585, 370, 245, 126]
10 scenarios: [585, 245]
11
12 # Choose whether to evaluate train and test data, and/or projections
13 on_train_test: False
14
15 # Data directory
16 dataset_root: data_kraken
17
18 # Input path
19 input_path: inputs
20
21 # Output path
22 output_path: outputs
23
24 # Variable
25 var_extr: tasmax
26
27 # Input Files
28 #
29 # Historical
30 histo_extr:
31 - tasmax_day_CMCC-ESM2_historical_r11p1f1_gn_19500101-19741231.nc
32 - tasmax_day_CMCC-ESM2_historical_r11p1f1_gn_19750101-19991231.nc
33 - tasmax_day_CMCC-ESM2_historical_r11p1f1_gn_20000101-20141231.nc
34
35 # Scenario-based projections
36 scenario_extr:
37 585:
38 - tasmax_day_CMCC-ESM2_ssp585_r11p1f1_gn_20150101-20391231.nc
39 - tasmax_day_CMCC-ESM2_ssp585_r11p1f1_gn_20400101-20641231.nc
40 - tasmax_day_CMCC-ESM2_ssp585_r11p1f1_gn_20650101-20891231.nc
41 - tasmax_day_CMCC-ESM2_ssp585_r11p1f1_gn_20900101-21001231.nc
42 370:
43 - tasmax_day_CMCC-ESM2_ssp370_r11p1f1_gn_20150101-20391231.nc
44 - tasmax_day_CMCC-ESM2_ssp370_r11p1f1_gn_20400101-20641231.nc
45 - tasmax_day_CMCC-ESM2_ssp370_r11p1f1_gn_20650101-20891231.nc
46 - tasmax_day_CMCC-ESM2_ssp370_r11p1f1_gn_20900101-21001231.nc
47 245:
```

Figure 14 Template for the main configuration of the input data to the DT



D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

```
65 max_val_loss: 0.0
66
67 TRAIN:
68 # Number of members used for training the network
69 n_memb: 1
70
71 # Initialize learning parameters
72 lr0: 0.001
73 batch_size: 10 # batch_size = 64 (alternative option)
74 epochs: 100 # epochs = 100 (alternative option)
75 strategy: ddp
76
77 # Early stopping parameters
78 # Under 1% improvement, consider the model starts converging
79 stop_delta: 0.01
80 # Wait for a few epochs to be sure before actually stopping
81 patience: 15
82 # Count when validation loss < stop_delta
83 early_count: 0
84 # Keep track of validation loss at t-1
85 old_valid_loss: 0.0
86
87 # Random high value for validation loss initialization
88 min_valid_epoch_loss: 100
89
90 MODEL:
91 # Kernel size (4,4)
92 kernel_size: 4
93 # Initial number of filters
94 init_channels: 8
95 # Number of input channels/variables (e.g., max temperature, precipitation, wind)
96 image_channels: 2
97 # Latent space dimension (in which the image is compressed)
98 latent_dim: 128
99
100 # KL divergence handles dispersion of information in latent space
101 # A balance is to be found with the prevailing reconstruction error
102 beta: 0.1
103
104 # Number of evaluations for each dataset
105 n_avg: 20
106
107 # Anomaly detection threshold (percentile)
108 anomaly_percent: 99
109
```

Figure 15 Template for the configuration of Training and Model configurations and parameters

```
107
110 # Training workflow
111 training_pipeline:
112   _target_: itwinai.pipeline.Pipeline
113   steps:
114     preprocessing-step:
115       _target_: itwinai.plugins.xtclim.preprocessing.preprocess_functions_2d_ssp.PreprocessData
116       dataset_root: ${GENERAL.dataset_root}
117       input_path: ${GENERAL.input_path}
118       output_path: ${GENERAL.output_path}
119       histo_extr: ${GENERAL.histo_extr}
120       landsea_mask: ${GENERAL.landsea_mask}
121       min_lon: ${GENERAL.min_lon}
122       max_lon: ${GENERAL.max_lon}
123       min_lat: ${GENERAL.min_lat}
124       max_lat: ${GENERAL.max_lat}
125       scenarios: ${GENERAL.scenarios}
126       scenario_extr: ${GENERAL.scenario_extr}
127     preprocessing-split-step:
128       _target_: itwinai.plugins.xtclim.preprocessing.preprocess_2d_seasons.SplitPreprocessedData
129       input_path: ${GENERAL.input_path}
130       n_memb: ${TRAIN.n_memb}
131       scenarios: ${GENERAL.scenarios}
132     training-step:
133       _target_: itwinai.plugins.xtclim.src.trainer.TorchTrainer
134       input_path: ${GENERAL.input_path}
135       output_path: ${GENERAL.output_path}
136       seasons: ${GENERAL.seasons}
137       epochs: ${TRAIN.epochs}
138       lr: ${TRAIN.lr0}
139       batch_size: ${TRAIN.batch_size}
140       n_memb: ${TRAIN.n_memb}
141       beta: ${MODEL.beta}
142       n_avg: ${MODEL.n_avg}
143       stop_delta: ${TRAIN.stop_delta}
144       patience: ${TRAIN.patience}
145       # Model parameters
146       kernel_size: ${MODEL.kernel_size}
147       init_channels: ${MODEL.init_channels}
148       image_channels: ${MODEL.image_channels}
149       latent_dim: ${MODEL.latent_dim}
150     inference-step:
151       _target_: itwinai.plugins.xtclim.src.trainer.TorchInference
152       input_path: ${GENERAL.input_path}
153       output_path: ${GENERAL.output_path}

```

Figure 16 Template for the description of the DT workflow with parameters for each step

3.1.3 Preconditions

Users have access to DT input data (CMIP6). No pre-trained ML model is provided because it depends on specific users' choices (climate model, region, atmospheric variable). The end user is expected to have basic python level knowledge about notebooks, and also to understand climate model simulation datasets.

3.1.4 Validation and Results

The DT application has been validated using 2m Daily Maximum Temperature input data from one GCM and several SSPs²⁷. First, the loss function was analyzed over the training and the test period, to ensure its stability. The inference results for several SSPs were compared to results of the percentile-based climate indices resulting from a classical analysis (*icclim*²⁸ software which is not using AI). The loss value is correlated to the percentage of anomalies (extremes) in the analyzed atmospheric variable.

Examples of output can be seen in both **Figure 17** and **Figure 18**. Those results can be used to study the trends in climate extremes over the regions and variables of interest.

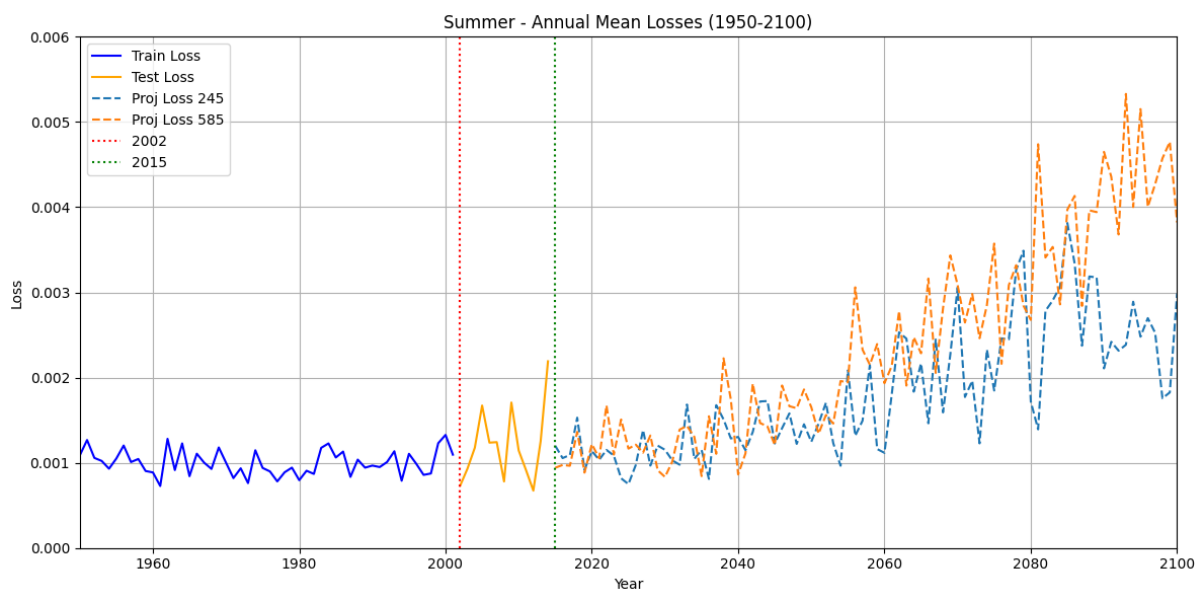


Figure 17 Annual Mean Losses for the 2m Daily Maximum Temperature

²⁷ <https://gmd.copernicus.org/articles/9/3461/2016/>

²⁸ <https://icclim.readthedocs.io/en/stable/>

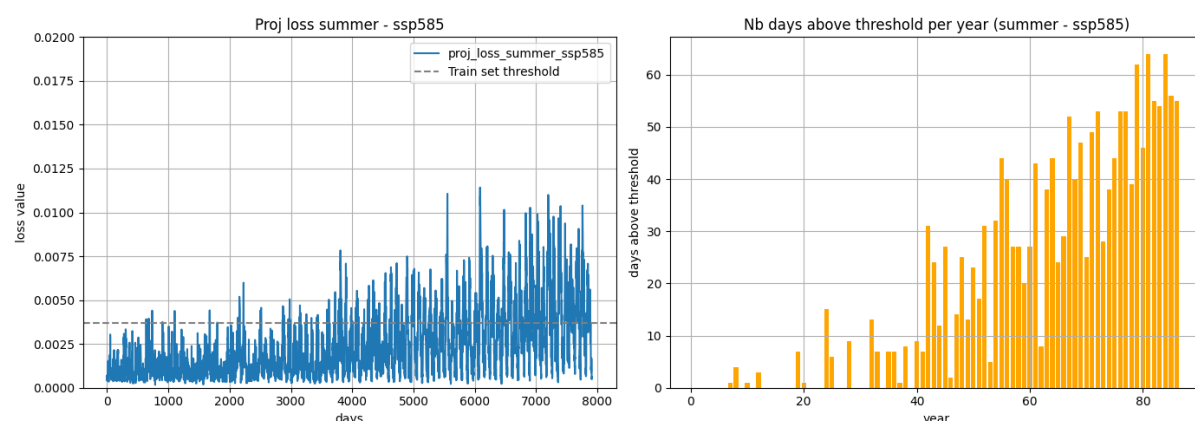


Figure 18 Number of days above threshold anomaly detection per year for the 2m Daily Maximum Temperature. Those anomalies represent climate extremes (very hot days)

3.2 Wildfire danger prediction in response to climate change

3.2.1 DTs Application Integrations

Model requirements

The primary objective of the wildfires DT case study is to design DL architectures, inspired by CNNs, that can capture complex relationships between selected input variables and burned areas at a global scale. These models produce outputs in theoretical percentage hectares of burned area, providing an indication of wildfire severity across different regions of the world.

The ML model uses as input data a subset of climate and environmental variables from the SeasFire Cube datacube for training (see section 2.2 for more details). Among the fire drivers provided in the SeaFire dataset, **Table 2** lists the fire predictors for which we have trained our UNet++ model to demonstrate the use case. The SeasFire Cube data can be downloaded from Zenodo²⁹.

Table 2 SeasFire Cube and corresponding CMIP6 data variables identified to carry out the wildfires prediction case study

Full name	SeasFire Cube name	Unit	CMIP6 name
ERA5 Reanalysis Data			
Total precipitation	tp	m	pr
Relative humidity	rel_hum	%	hur

²⁹ SeasFire Cube: <https://zenodo.org/records/8055879>

D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

Temperature at 2 meters – Min	t2m_min	K	tasmin
Land-Sea mask	lsn	0-1	sftlf
Nasa MODIS MOD11C1, MOD13C1, MCD15A2			
Leaf Area Index	lai	m2m-2	lai
Global Wildfire Information System (GWIS)			
Burned Areas from FCCI	fcci_ba	ha	Used only for training

The trained model can be applied either on SeasFire cube data or CMIP6 data. A RUCIO “dataset” in the interTwin Data Lake called *ScenarioMIP* (under the CMIP6 scope) has been created with the following variables at daily resolution and 100 km scale: *sftlf*, *hur*, *lay*, *sfcWinds*, *pr*, *tasmin* and *tasmax* (more details on the data are reported in section 2.2). **Table 3** shows a breakdown of the different model and climate scenarios considered for the DT application demonstration. Such data can be accessed by simply querying the interTwin Data Lake infrastructure and by other DT applications (i.e., generic detection of climate extremes). The ML4Fires thematic module supports direct access to data on the interTwin Data Lake based on RUCIO. Additional CMIP data can be downloaded from ESGF³⁰ and uploaded to the interTwin Data Lake using the *esgpull_rucio* (WP7) toolkit from the DTE.

Table 3 CMIP6 data from ScenarioMIP project made available on the interTwin Data Lake

Model name	Scenarios available	Volume
CMCC-ESM2	ssp126, ssp245, ssp585	167GB
NorESM2-MM	ssp126, ssp245	138GB
CESM2	ssp126, ssp245, ssp585	256GB
MPI-ESM1-2-HR	ssp126, ssp245, ssp585	316GB

³⁰ ESGF Search portal at IPSL: <https://esgf-node.ipsl.upmc.fr/search/cmip6-ipsl/>



D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

In the use case of wildfires, distinction is not made between the cause of the fire. That is to say that burned areas caused by human intervention (directly - such as arson - or indirectly caused due power grids etc.) or natural (lightning).

Furthermore, for stable model learning, the target variable is not directly the area burned hectares but instead the theoretical percentage of the pixel burned to forest fires. This limits the range of the output value between 0 and 1.

Workflow

Figure 19 depicts the high level workflow related to the DT application on burned areas prediction and the links with the project components and infrastructure.

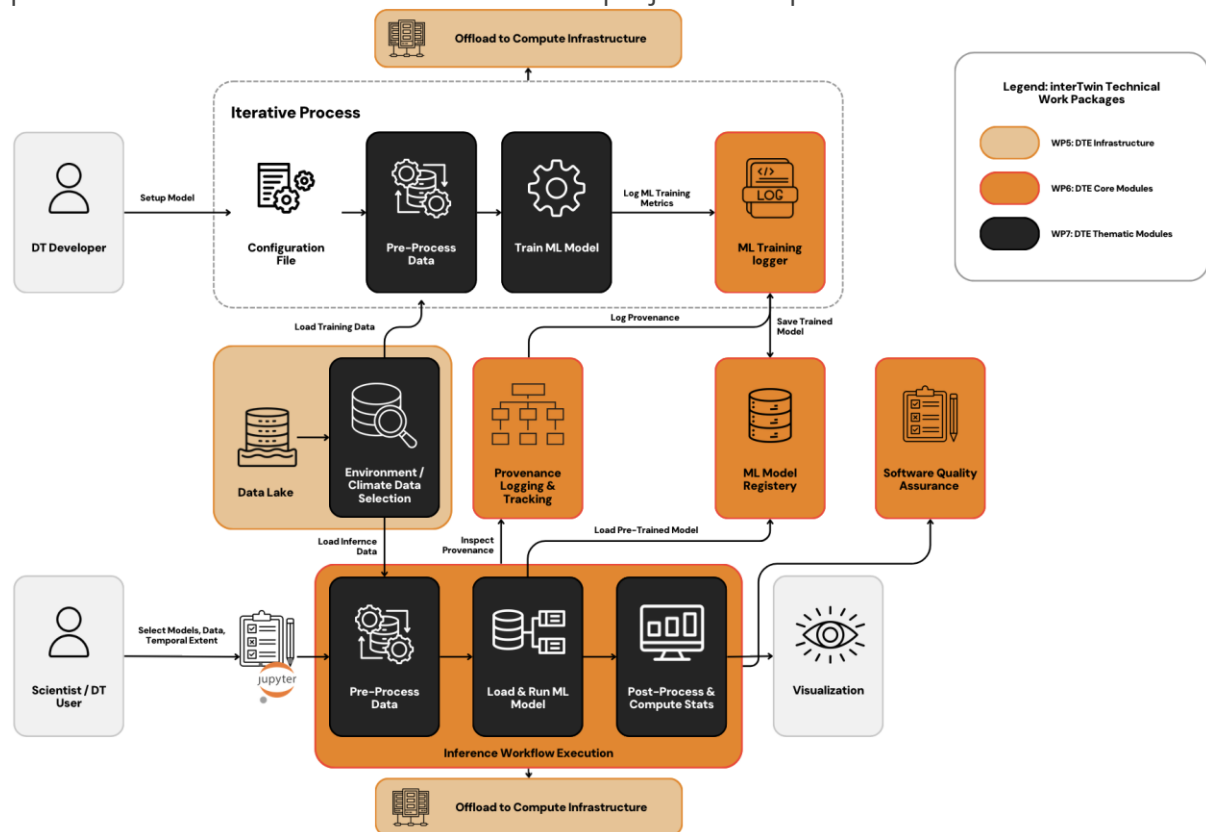


Figure 19 Overview of the wildfires DT application workflow

In particular, wildfires DT exploits the following components from the project:

- WP5:
 - **RUCIO**: All the data needed by the DT, both for training and inference is available on RUCIO. CMIP6 data can be retrieved using the RUCIO client;
 - **interLink**: The DT application relies on interLink for deploying the Singularity images and offloading the computation on the project testbed infrastructure (e.g., Vega);
- WP6:



- **PyOphidia and yProv:** Workflow management system (i.e., PyOphidia) for running the inference pipelines. A PyOphidia extension also allows tracking the workflow provenance that can be inspected with yProv components;
- **itwinai and yProv4ML:** Skills and provenance of an ML model can be tracked respectively on MLflow and via yProv4ML during training through the *itwinai* framework. The same interface is also used to log artifacts on MLflow (e.g., the model weights, the provenance documents)
- **SQAaaS platform:** In particular the Ophidia Workflow Validation Tool is used to validate the PyOphidia workflow structure;
- WP7:
 - **Esgpull_rucio:** can be used for gathering data from ESGF nodes and uploading them to RUCIO;
 - **ML4Fires:** the DT application heavily relies on the modules and libraries from the *ML4Fires* thematic module for training a ML model, validating the results and running the inference. The notebooks are stored in the *notebooks* folder the module³¹.

The inference pipeline, based on an Ophidia workflow, allows for production of burned areas results across an ensemble of different climate models³². This includes multiple Ophidia tasks covering the whole data preparation, ML model execution and data post-processing steps. In particular, given a set of input parameters (see next section), the workflow takes care of selecting the variables from the input NetCDF files, aggregating them accordingly (different operations can be applied to the various variables), regridding the variables to a common grid, running the provided pre-trained ML model on such data, masking the results and computing the statistics from the ensemble of results on multiple CMIP6 models. The workflow is implemented in cwl/json and can be managed with the PyOphidia module. Provenance information can also be produced by the module, once the execution is completed.

3.2.2 Scope and limitations

The DT application serves a wide variety of professionals in terms of the level of expertise (climate/environmental scientists). Given the wide range of end-users, the DT strived to strike a balance between simplicity (for less technically inclined uses) and high configurability (for expert users). Therefore, the use of a familiar Jupyter Notebook makes it easy for non-technical end-users (DT application users) and the TOML³³ based configuration provides high configurability for the expert users (DT application developers). For example, the developer can set the key hyperparameters (e.g., training metrics, loss, driver list, etc.).

³¹ https://github.com/CMCC-Foundation/ML4Fires/tree/main/digital_twin_notebooks

³² <https://github.com/CMCC-Foundation/ML4Fires/tree/main/workflows>

³³ <https://toml.io/en/>



The notebooks are used for validating the trained ML model and to perform projection of the burned areas using the CMIP6 dataset. Scripts are provided for running the training process.

The Jupyter Notebook allows to:

1. Select through widgets:
 - a. CMIP6 model and scenario from a range of models and scenarios available (see **Table 3**).
 - b. The years to include in the analysis (from the range 2015-2100);
 - c. ML model from a set of pre-trained models using the experiments stored in the MLflow framework;
 - d. The aggregations to be performed - on a monthly/seasonal/annual basis - on inferred burned areas from CMIP6 data. Furthermore, the statistics of the aggregation can also be selected.
2. Run the DT workflows on the selected input data and pre-trained model;
3. Save as NetCDF and/or visualise the results. Different indicators can be provided, for example:
 - a. seasonal/annual burned area maps
 - b. burned areas aggregated by region

Figure 20 and **Figure 21** show the interface of the notebook demonstrating these options. **Figure 22** shows results from different aggregation schemes for the 8-day burned area predictions: in the left image, the burned areas were summed on a monthly scale and then averaged on yearly and decadal scales, while in the right image mean was taken on monthly and then on yearly and decadal scales. These results are computed on the CMIP6 model CMCC-ESM2 for the scenario SSP126.

D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

```
import ipywidgets as widgets

scenario = widgets.Dropdown(
    options=[('SSP126', 'ssp126'), ('SSP245', 'ssp245'), ('SSP370', 'ssp370'),
            ('SSP585', 'ssp585')],
    value = 'ssp126',
    style={'description_width': '60px'},
    description='Scenario', disabled=False,
    layout=widgets.Layout(width='200px'))

climate_model = widgets.Dropdown(
    options=[('MPI-ESM1-2-HR', 'MPI-ESM1-2-HR'), ('CMCC-ESM2', 'CMCC-ESM2'), ('NorESM2-MM', 'NorESM2-MM'),
            ('CESM2', 'CESM2')],
    value = 'CMCC-ESM2',
    style={'description_width': '60px'},
    description='Model', disabled=False,
    layout=widgets.Layout(width='200px'))

year_range = widgets.IntRangeSlider(
    value=[2030, 2035],          # initial range
    min=2015,                   # min value
    max=2100,                   # max value
    step=1,                     # step size
    description='Year range:',
    style={'description_width': '80px'},
    layout=widgets.Layout(width='400px')
)

display(widgets.HBox([scenario, climate_model, year_range]))
```

Scenario Model Year range:

Figure 20 Interface to choose the CMIP6 models and scenarios along with the years to include in the analysis.




```
monthly_agg = widgets.Dropdown(
    options=[('Mean', 'mean'), ('Sum', 'sum'), ('Median', 'median'),
            ('Mode', 'mode')],
    value = 'sum',
    style={'description_width': '150px'},
    description='Monthly aggregate', disabled=False,)

monthly_add_period = widgets.Dropdown(
    options=[('1 Month', '1M'), ('2 Months', '2M'), ('4 Months', '4M'),
            ('6 Month', '6M')],
    value = '1M',
    style={'description_width': '150px'},
    description='Monthly aggregate period', disabled=False,)

yearly_agg = widgets.Dropdown(
    options=[('Mean', 'mean'), ('Sum', 'sum'), ('Median', 'median'),
            ('Mode', 'mode')],
    value = 'mean',
    style={'description_width': '150px'},
    description='Yearly aggregate', disabled=False,)

decadal_agg = widgets.Dropdown(
    options=[('Mean', 'mean'), ('Sum', 'sum'), ('Median', 'median'),
            ('Mode', 'mode')],
    value = 'mean',
    style={'description_width': '150px'},
    description='Decadal aggregate', disabled=False,)

display(widgets.HBox([monthly_agg,
                      monthly_add_period]))

display(widgets.VBox([yearly_agg,
                      decadal_agg]))
```

Monthly aggregate	Sum	▼	Monthly aggregate per...	1 Month	▼
Yearly aggregate	Mean	▼			
Decadal aggregate	Mean	▼			

Figure 21 Interface to choose the aggregation of the 8-day burned area forecast on the seasonal, yearly, and decadal time-scale.

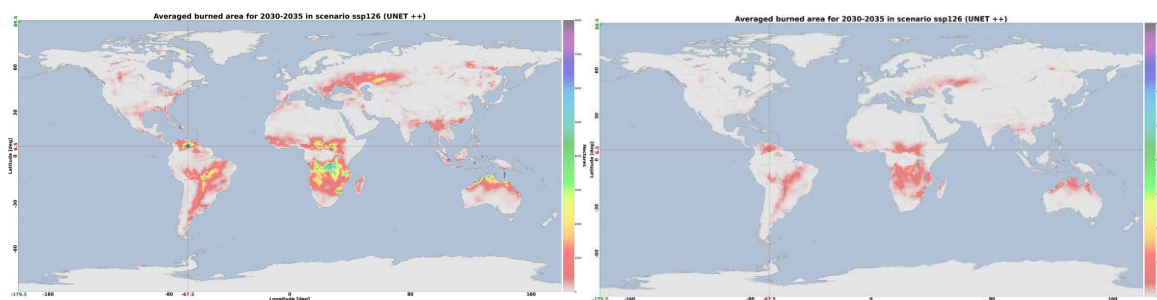


Figure 22 Results from different aggregation schemes for the 8-day burned area predictions applied on the CMIP6 model CMCC-ESM2 for the scenario SSP126

3.2.3 Preconditions

Users have access to DT data (i.e., SeasFire cube, CMIP6), pre-trained ML models, thematic components and Jupyter notebook, as well as the interTwin platform. The end-user is

expected to have basic level knowledge of how to work with Python-based notebooks.

3.2.4 Validation and Results

In order to validate the ML model training process we track the loss values on MLflow, i.e. the weighted binary cross-entropy for each epoch, to make sure there is no overfitting. Then we further validate the trained model on a test set i.e., years 2019 and 2020 of the SeasFire cube. These years have not been used as part of the training or validation but instead on dataset that the network has never seen. Different metrics including Mean Absolute Error (MAE) and Structural Similarity Index Measure (SSIM) are computed to quantify how far the predictions are with respect to ground truth.

To visually validate the results a map with the difference between the prediction and ground truth, in terms of percentage of burned area per grid point, is also computed and averaged across the test set time range (see **Figure 23**). As it can be seen differences are in the range of $\pm 0.2\%$. The resulting MAE value on the same test set is 0.01 % while the similarity index (SSIM) is 0.95. To simplify the validation phase a Jupyter Notebook is available³⁴.

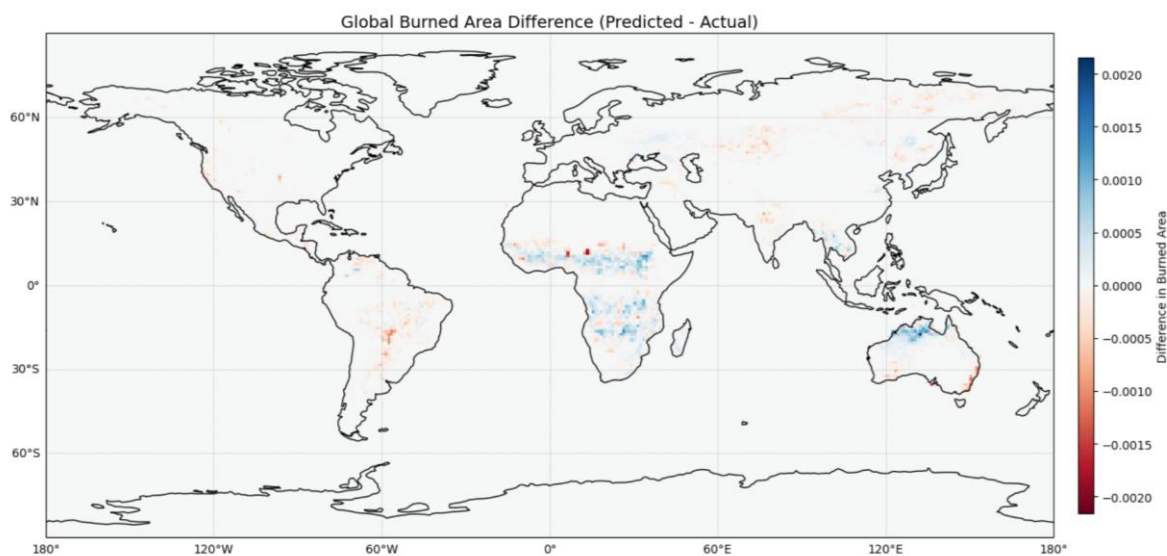


Figure 23 Average difference in terms of percentage of burned area per grid point on the test set between the values predicted by the ML model and the ground truth from the SeasFire cube

³⁴ https://github.com/CMCC-Foundation/ML4Fires/tree/main/digital_twin_notebooks

3.3 Tropical storms change in response to climate change

3.3.1 DTs Application Integrations

Model requirements

The ML model uses as input data a subset of climate and environmental variables from ERA5, combined with IBTrACS information to be used during the supervised training. The portion of ERA5 data is divided as follows:

- Training: 30 years — from 1980 to 2009 — 70% of the data
- Validation: 4 years — from 2010 to 2013 — 10% of the data
- Testing: 8 years — from 2014 to 2021 — 20% of the data

As reported in **Table 4**, this data occurs with a 6-hour temporal resolution and a 0.25-degree spatial resolution. The set of variables considered for training is reported in **Table 4**. The set of variables considered in the final implementation of the ML model includes only psl and vo_850.

Table 4 Input parameters to the ML model

Full name	CMIP6 name	ERA5 name	Unit
10m wind gust since previous post-processing	N.A.	fg10	m/s
instantaneous 10m wind gust	N.A.	i10fg	m/s
temperature at 500 mb	tas (at pressure level 500 hPa)	t_500	K
temperature at 300 mb	tas (at pressure level 300 hPa)	t_300	K
relative vorticity at 850 mb	Rv850 (when available)	vo_850	1/s
mean sea level pressure	psl	msl	Pa

The trained model can be applied either to ERA5 data or to high-resolution CMIP6 data. In particular, data from the HighResMIP³⁵ [10] CMIP6 project has been uploaded on the interTwin Data Lake and the related “dataset” called *HighResMIP* has been created under the CMIP6. In particular three variables have been considered from CMIP6 (i.e., *ua*, *va* and *psl*) at a 25 km scale and 6 hourly resolutions. In this case only the ssp585 future emission scenario (called “highres-future”) is available. **Table 5** shows a breakdown of the different models considered for the DT application demonstration. Such data has been then processed before being fed to the ML model by using an Ophidia workflow³⁶. The workflow takes care of extracting the use case spatial area, compute the relative vorticity from the wind components (*ua*, *va*) and performing other data manipulation operations to make the data more readily usable by the ML model. As the original data volume is quite large (about 4TBs), the workflow has been executed only once and the resulting processed data (around 400GBs) is stored on the interTwin Data Lake. Such data can then be directly accessed by querying the Data Lake infrastructure based on RUCIO.

Table 5 CMIP6 data from HighResMIP project made available on the interTwin Data Lake

Model name	Scenarios available	Volume (original)
CMCC-CM2-VHR4	ssp585	1.9TB
EC-Earth3P-HR	ssp585	1.1TB
MPI-ESM1-2-HR	ssp585	344GB
CNRM-CM6-1-H	ssp585	515GB

Workflow

Figure 24 shows the high-level view of the workflow related to the digital twin use case on TCs detection and links with the project components/infrastructure.

³⁵ https://highresmip.org/experiments/experiment_cmip6/

³⁶ <https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection/tree/main/workflows>



D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

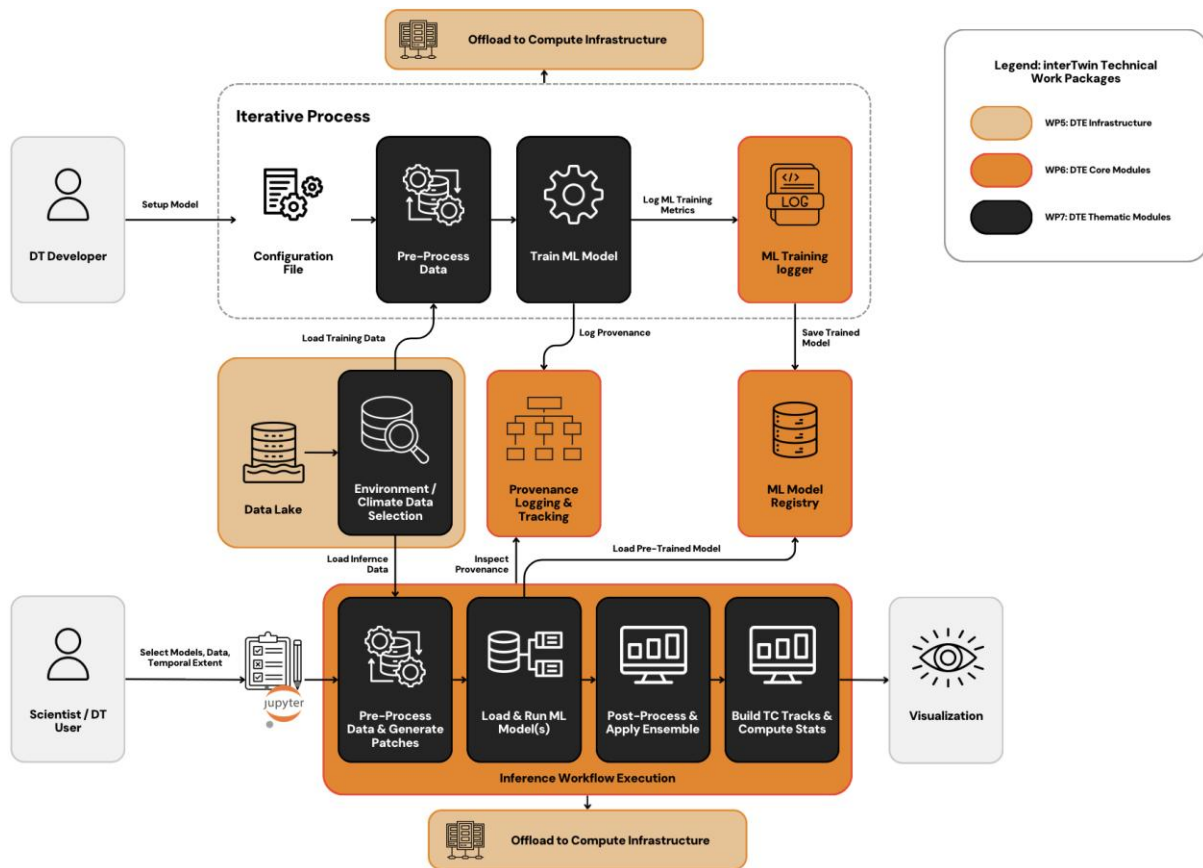


Figure 24 Overview of the Tropical Cyclones DT application

In particular, the tropical cyclones DT application exploits the following components from the project:

- WP5:
 - **RUCIO**: Data for training and inference can be accessed from the interTwin Data Lake based on RUCIO;
 - **interLink**: similarly to the previous case, the DT application relies on interLink for deploying the Singularity images and offloading the computation on the project testbed infrastructure (e.g., Vega);
- WP6:
 - **PyOphidia and yProv**: a workflow management system (i.e., PyOphidia) has been used for pre-processing the CMIP6 data and preparing it to be used by the pre-trained ML models. This allows also to track the workflow provenance that can be inspected with yProv components;
 - **itwinai and yProv4ML**: *itwinai* logger is used to track the model performance during the training on MLflow as well as the provenance by relying on yProv4ML. Itwinai is also used to store the training artifacts on MLflow;
 - **SQAaaS platform**: The Ophidia Workflow Validation Tool can be used to validate the PyOphidia workflow structure;

- WP7:
 - **ML TC Detection:** the workflow depends on the modules available in the *ML TC Detection* thematic module for running the functions for training the different implementations of the ML model, validating the results. The module also provides a feature for running the overall pipeline for tracking TCs on climate data. The Jupyter Notebooks use the functions made available by the library.

3.3.2 Scope and limitations

The goal is to provide Jupyter Notebooks for expert users (climate/environmental scientists) to:

The Jupyter Notebook³⁷ allows:

1. Select:
 - a. CMIP6 model from a range of models available (see **Table 5**).
 - b. The years to include in the analysis (from the range 2015-2050);
 - c. ML model from a set of pre-trained models using the experiments stored in the MLflow framework;
2. Run the DT workflows on the selected input data and pre-trained models;
3. Save as CSV and/or visualise the results. Different indicators can be provided, for example:
 - a. TC tracks and number of tracks for each year/month;
 - b. Duration of TCs per year.

Figure 25 shows the interface of the notebook for selecting these options, while **Figure 26** displays the results of the application of the TC tracking workflow applied on a subset of the HighResMIP model CMCC-CM2-VHR4.

³⁷ https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection/blob/main/notebook/inference_notebook.ipynb




```
import ipywidgets as widgets

climate_model = widgets.Dropdown(
    options=[('CNRM-CM6-1-HR', 'CNRM-CM6-1-HR'), ('CMCC-CM2-VHR4', 'CMCC-CM2-VHR4'), ('EC-Earth3P-HR', 'EC-Earth3P-HR'),
            ('MPI-ESM1-2-HR', 'MPI-ESM1-2-HR')],
    value = 'CMCC-CM2-VHR4',
    style={'description_width': '120px'},
    description='Model', disabled=False,
    layout=widgets.Layout(width='300px'))

year_range = widgets.IntRangeSlider(
    value=[2030, 2035],      # initial range
    min=2015,                # min value
    max=2050,                # max value
    step=1,                  # step size
    description='Year range:',
    style={'description_width': '80px'},
    layout=widgets.Layout(width='400px'))

)

display(widgets.HBox([climate_model, year_range]))
```

Model: Year range:

Figure 25 Interface to choose the CMIP6 models along with the years to include in the analysis.

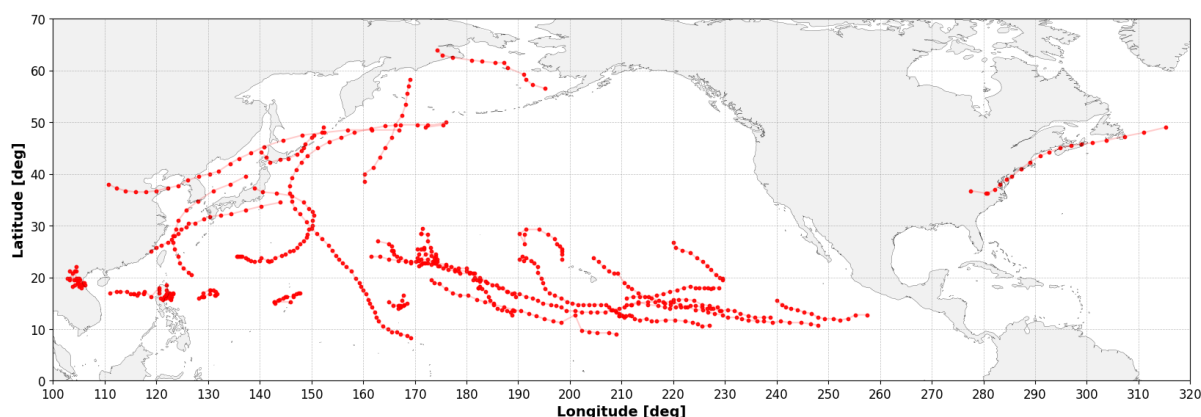


Figure 26 Results from the TC tracking applied on a subset of the CMIP6 model CMCC-CM2-VHR4

3.3.3 Preconditions

Users have access to DT data, pre-trained ML models, thematic components and Jupyter notebook, as well as the interTwin platform.

3.3.4 Validation and Results

To evaluate the generalization capabilities of the models, their performance is assessed on out-of-samples and compared against IBTrACS values. Two widely used evaluation metrics are adopted, Probability of Detection (POD) and False Alarm Rate (FAR), which are commonly employed to assess the accuracy of TC tracking systems. These metrics are defined as follows:

$$POD = \frac{H}{H + M}$$

$$FAR = \frac{FA}{H + FA}$$



Where H is the number of observed tracks correctly matched by at least one detected track during the TC lifetime, M is the number of observed tracks with no corresponding detected tracks and FA is the number of detected tracks that don't match any observed track. The final setup of the model has a POD of 84.2% and a FAR of 28.2%. **Figure 27** shows an example of the TC track ("Keoni" cyclone) detected by our final setup and compares it with the actual track available in the IBTrACS dataset.

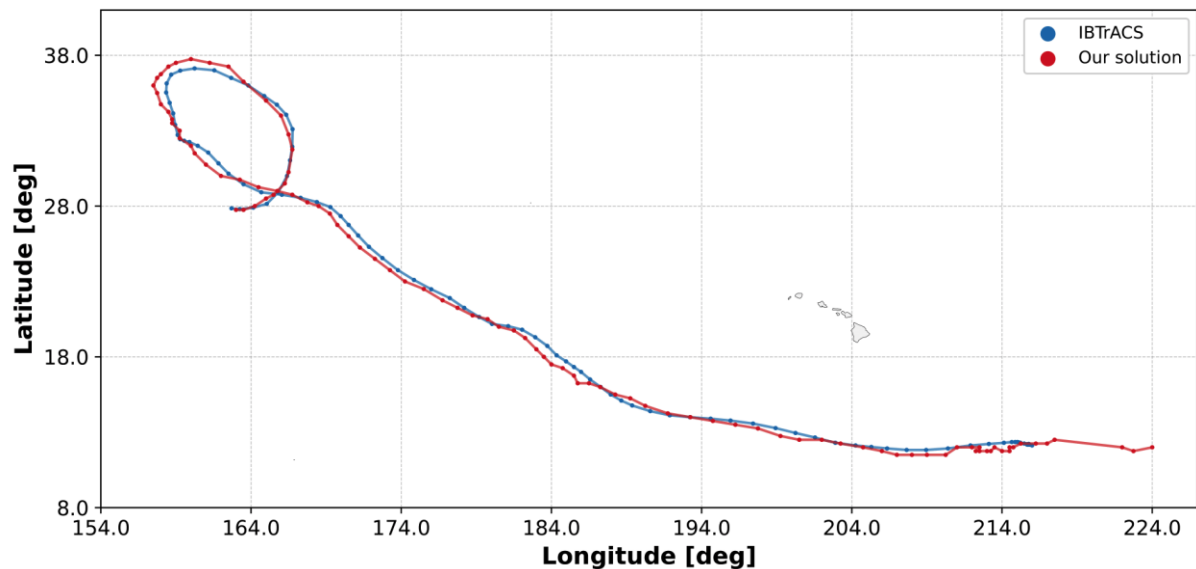


Figure 27 IBTrACS (in red) and detected TC trajectory (in blue) for the Keoni tropical cyclone

3.4 Eddies detection

The Eddies DT Application addresses oceanic mesoscale eddies analysis. It can be considered an example of "exploitation" DT, as it applies interTwin technologies to an extra DT provided by an external institution; in particular the ML pipeline comes from the Alfred Wegener Institut (AWI)³⁸ and it has been ported, tested and validated both on VEGA EuroHPC and the ENES Data Space infrastructure³⁹. Porting includes also transferring the output of FESOM2⁴⁰ simulations by downloading them from the AWI storage. The Eddies DT repository includes the script to perform the main steps of the pipeline, including interpolation of FESOM2 unstructured grids to matrices, production of the segmentation masks with pyEddyTracker⁴¹, training of CNN models and inference.

³⁸ <https://www.awi.de/en/>

³⁹ <https://enesdataspace.vm.fedcloud.eu/>

⁴⁰ <https://fesom.de/models/fesom20/>

⁴¹ <https://py-eddy-tracker.readthedocs.io/en/stable/>

3.4.1 DTs Application Integrations

Model requirements

The ML model uses as input data from AWI. In particular the input dataset consists of 132 files (1 file per month) and it covers a 10-year timeframe from 2009 to 2019, with daily temporal resolution and 1/12° spatial resolution. Overall the entire dataset is about 35GB. The focus is on the South Atlantic (70W to 30E and 60S to 20S) which represents an Eddy-rich region.

- Training: 7 years, from 2009 to 2015 — 64% of the data
- Testing: 2 years, from 2016 to 2017 — 18% of the data
- Validation: 2 years from 2018 to 2019 — 18% of the data

The variable used in the training was the SSH variable (Sea Surface Height).

Workflow

Figure 28 shows the high-level view of the workflow related to the Eddies DT use case.

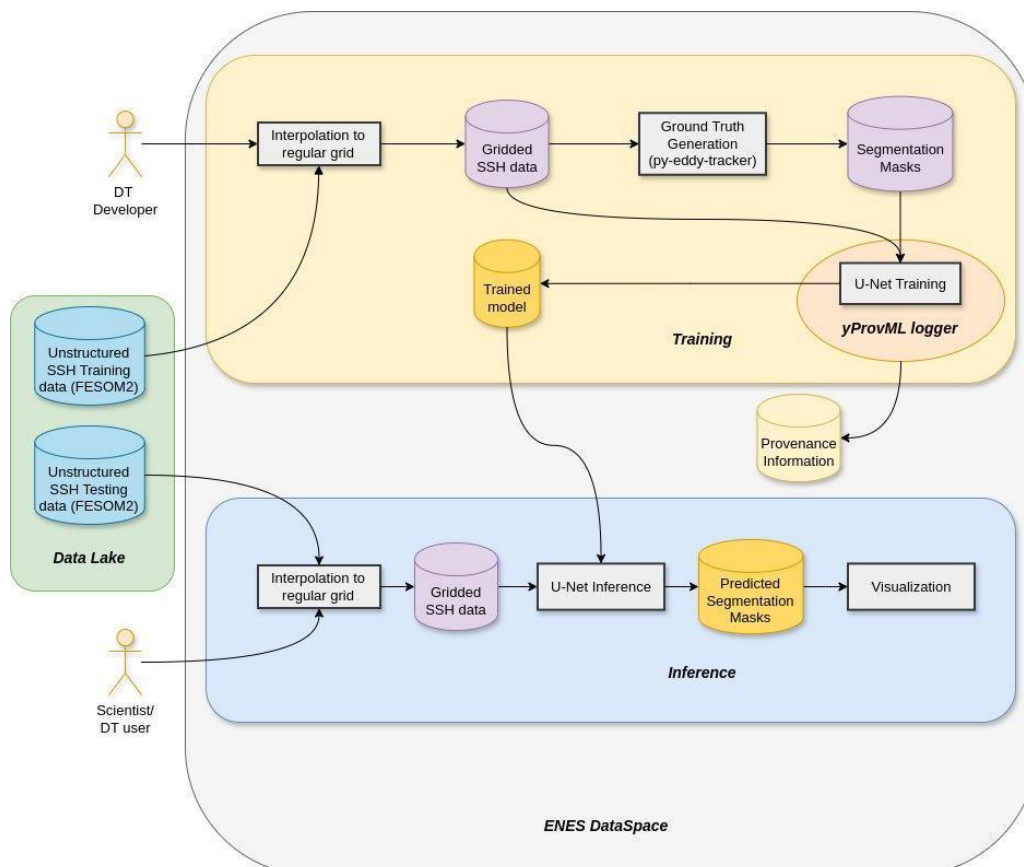


Figure 28 Eddies DT unified training and inference workflow

The workflow consists of two pipelines: training and inference. While the first one is managed by the DT developer, the second one applies to the DT user (e.g., scientist).

D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

The “DT training pipeline” starts from the SSH variable taken from the unstructured data generated by the FESOM2 model, it provides an interpolation on a regular grid and then it goes as input to the U-Net training jointly with the segmented data from the ground truth generation task. The traceability of the ML training process is ensured by the yProv4ML integration, which delivers the provenance document represented in **Figure 29** (code) and **Figure 30** (provenance graph).

The “DT inference pipeline”, takes as input testing data from FESOM2, it does (as before) an interpolation to a regular grid and then it uses the ML model to run the U-Net inference. As a result, the predicted segmentation masks are generated.

The ML steps are performed with the help of the TensorFlow framework. Concerning the links with the project components/infrastructure, an integration with both yProv (WP6) and the EddiesML component (WP7), which provides foundational modules for this workflow, was performed. The deployment was done on the ENES Data Space infrastructure for the preliminary experimental phase and on EuroHPC VEGA at a later stage to validate the final version of the DT.

```
In [2]: from tensorflow.python.client import device_lib
print(device_lib.list_local_devices())
import tensorflow as tf; print(tf.config.list_physical_devices('GPU'))

[Name: "/device:CPU:0"
 device_type: "CPU"
 memory_limit: 268435456
 locality {
 }
 incarnation: 17187469516857629593
 xla_global_id: -1
 , name: "/device:GPU:0"
 device_type: "GPU"
 memory_limit: 40353529856
 locality {
   bus_id: 4
   numa_node: 3
   links {
 }
 }
 incarnation: 17305422764933213044
 physical_device_desc: "device: 0, name: NVIDIA A100-SXM4-40GB, pci bus id: 0000:03:00.0, compute capability: 8.0"
 xla_global_id: 416903419
 ]
 [PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU')]
 2025-06-07 09:44:20.059803: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1928] Created device /device:GPU:0
 with 38484 MB memory: -> device: 0, name: NVIDIA A100-SXM4-40GB, pci bus id: 0000:03:00.0, compute capability: 8.
 0

In [3]: # Prov4ML initialization

import prov4ml

prov4ml.start_run(
    prov_user_namespace="test",
    experiment_name="torchless_run",
    provenance_save_dir="prov",
    collect_all_processes=False,
    save_after_n_logs=100
)

[codecarnon WARNING @ 09:44:20] Multiple instances of codecarnon are allowed to run at the same time.

...

In [28]: # Log trained model
prov4ml.log_model(model, "U-Net")

# Close yProvML
prov4ml.end_run(
    create_graph=True,
    create_svg=True,
)
```

Figure 29 EddiesML notebook integrating the AI provenance tracking capability. As it can be seen the integration with yProv4ML calls (cell 3 and 28) is lightweight and does not significantly impact on the user's code



D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

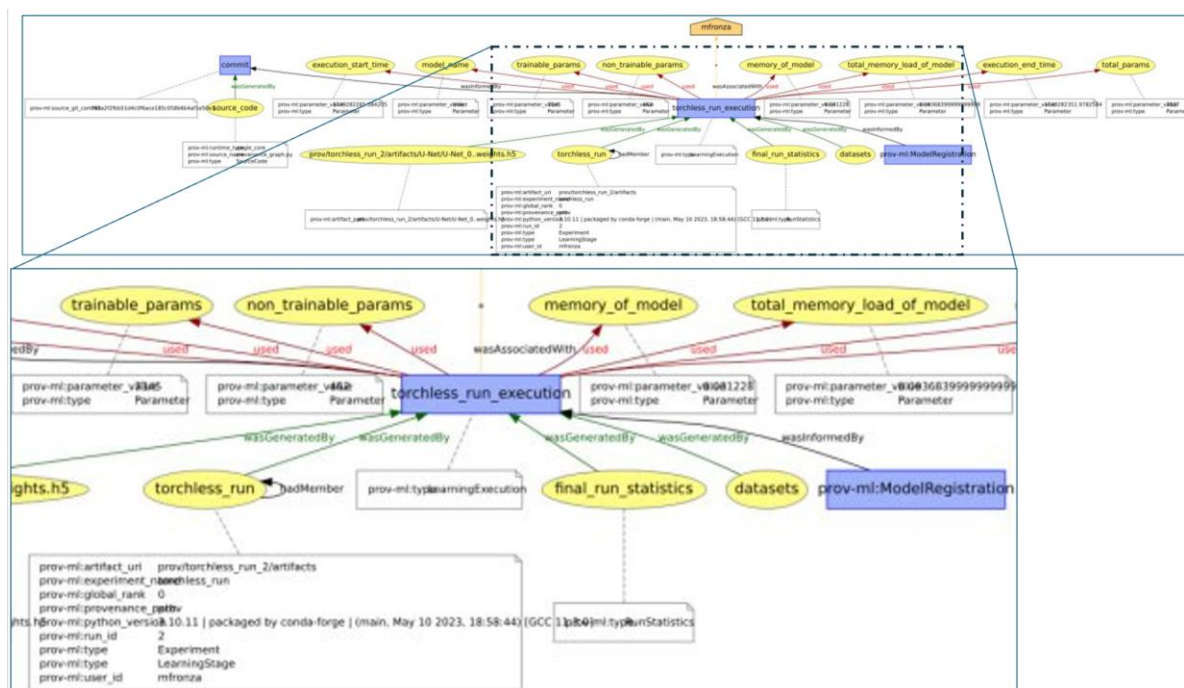


Figure 30 Traceability of the entire ML training process, documented via yProv4ML integration with a focus on a subset of the provenance graph, highlighting some of the metrics captured by the library at runtime

3.4.2 Scope and limitations

The goal is to provide Jupyter Notebooks for expert users (environmental scientists) in order to select the input data, spatial domain, temporal target, pre-trained ML model and then run the DT inference step. By properly configuring the input parameters in the Jupyter Notebook the users can then run the Eddies DT application to perform the ML task. A more general goal of this DT is that it can be considered an example of “exploitation” DT, as it applies, with limited effort, interTwin technologies to an extra DT provided by an external institution (AWI).

3.4.3 Preconditions

Users have access to DT data, pre-trained ML model, thematic components and Jupyter Notebook, as well as the interTwin platform.

3.4.4 Validation and Results

The accuracy metric for the loss function used has been the Sørensen–Dice coefficient, a statistic used to gauge the similarity of two samples.

According to its definition:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}$$

Where X represents the prediction set comprising all the pixels that a CNN identifies as belonging to an eddy, and Y is the ground truth set consisting of the pixels that correspond to an eddy.

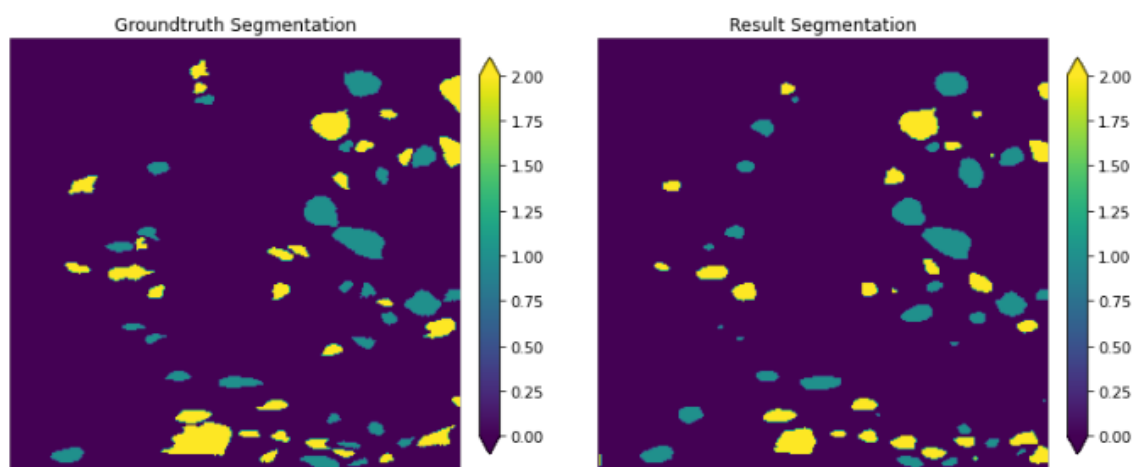


Figure 31 Validation phase pyEddyTracker vs Eddies DT inference

As shown in **Figure 31**, according to the validation performed during the last stage of the project, the overall accuracy was about 90% for the ground truth provided by the pyEddyTracker (two plots comparing the ground truth vs results segmentation are reported in the figure above).

In terms of performance, the run performed on the system showed an execution time on a single image (as “unit” of test) which was with the Eddies DT inference task 3 orders of magnitude faster than the pyEddyTracker application, showing the great impact in moving from a data-intensive (pyEddyTracker) approach to a data-driven one (Eddies DT).

3.5 Flood early warning in coastal and inland regions

This DT application leverages on the models and tools listed in the next section. These are open source and can be found at the respective repositories listed in D7.7⁴².

Jupyter Notebooks and CWL workflows are available at the project’s repositories and describe how to set up the necessary models and run them for the DT application leveraging the interTwin DTE.

3.5.1 DTs Application Integrations

Model requirements

The flood early warning DT for coastal and inland regions relies on two process-based

⁴² D7.7 Final version of the thematic module for the environment domain
<https://zenodo.org/records/14918025>

models combined with satellite observations of floods:

1. SFINCS: a reduced-complexity model designed for super-fast modelling of compound flooding in a dynamic way.
2. HydroMT: an open-source Python package that facilitates the process of building and analysing spatial geoscientific models with a focus on water system models. It does so by automating the workflow to go from raw data to a complete model instance which is ready to run and to analyse model results once the simulation has finished.
3. Satellite-based flood monitoring using dask-flood-mapper: an existing workflow for flood monitoring has been re-developed as a Dask-powered Docker container, ensuring scalability and interoperability on several platform backends. Publicly available datasets stored at the EODC facilitate the Dask implementation of Sentinel-1 based flood maps. These datasets are published as SpatioTemporal Asset Catalogues (STAC⁴³) — a common language to describe geospatial information, which ensures discoverability and interoperability of the data. The STAC catalogues can be accessed with common HTTP verbs, as well as via a variety of Python clients. The dask-flood-mapper implementation makes use of three datasets (or “collections” in STAC terminology): 1) the σ_0 backscatter data projected in Equi7Grid at 20 meter pixel spacing, 2) the Projected Local Incidence Angle (PLIA) values of those measurements, and 3) the harmonic parameters of a model fit on the pixel’s backscatter time series over land. This harmonic model fitted on historical data describes typical seasonal Sentinel-1 σ_0 backscatter variation on a 20 meter pixel level. These three collections of the STAC catalogue are respectively designated as: “SENTINEL1_SIG0_20M”, “SENTINEL1_HPAR”, and “SENTINEL1_MPLIA”. The dask-flood-mapper package also provides the option to re-calculate harmonic parameters for a particular region on-the-fly using more up-to-date backscatter data, which can provide for a better result at the cost of processing speed.

⁴³ <https://stacspect.org/en/>



Workflow

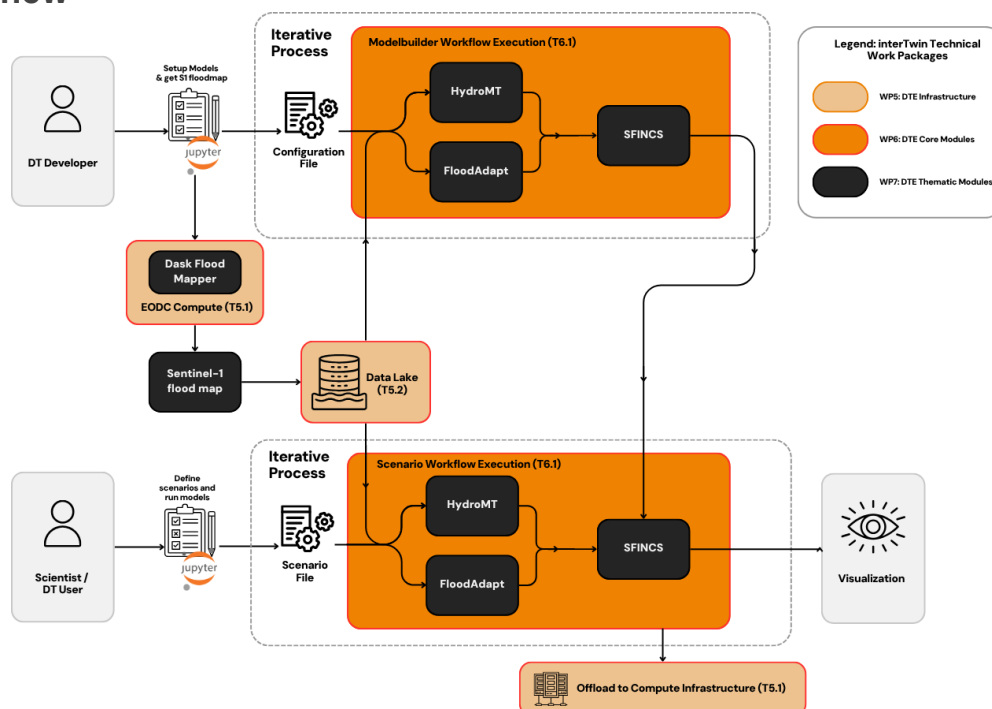


Figure 32 High-level workflow diagram for the flood early warning DT

The workflow for the flood early warning DT is implemented in a Jupyter Notebook (Figure 12 in the **Section on the flood climate impact DT**) and has been exploiting the following components from the project:

- WP7: The workflow depends on functionalities from the hydrological model data processing thematic module dependent on HydroMT⁴⁴, SFINCS⁴⁵, and dask-flood-mapper⁴⁶ to generate the flood maps.
- WP6: CWL workflows have been developed integrating SFINCS, HydroMT, and dask-flood-mapper to preprocess the SFINCS model, generate floodmaps and provide indicators for the comparison between the SFINCS and dask-flood-mapper outputs. The SFINCS model is run using the WP6 component OSCAR which offloads the job to HPC using interLink (WP5).
- WP5: Global data for model building are accessed from the interTwin Data Lake using RUCIO. To further improve the inundation model, the use case used local data for model building and preprocessing. It is up to the user to provide such local data, with an example provided in the notebook interface on how to configure such data for use in the DT.

3.5.2 Scope and limitations

The goal is to provide Jupyter Notebooks for expert users to:

⁴⁴ <https://www.intertwin.eu/article/thematic-module-hydromt-sfincs>

⁴⁵ <https://www.intertwin.eu/article/thematic-module-sfincs>

⁴⁶ <https://www.intertwin.eu/article/thematic-module-dask-flood-mapper>

1. Set up the necessary models for a user-defined region of interest.
2. Run the necessary models and Earth Observation processing pipelines to produce deterministic flood maps for a user-defined region of interest, validate the resultant output data against observations, and compare flood maps derived from satellite observations and process-based models.

A demonstration is provided for the Darss peninsula in northeast Germany.

The main limitation for Sentinel-1 based flood maps are the availability of suitable overpasses of the satellite. This means that flood events, or the peak flooding extent of an event, can be missed.

3.5.3 Preconditions

Users have access to DT data, models, thematic components and Jupyter Notebooks.

1. Users can:
 - a. Specify a region of interest,
 - b. Specify a temporal period to simulate,
 - c. Select local data for the models if available.
2. Users run the DT workflows for the specified region and period using default global data or selected local data if available.
3. The output of the DT can be visualised in the Jupyter Notebooks and the data can be downloaded/saved as NetCDF or GeoTIFF data.

3.5.4 Validation and Results

The SFINCS inundation model for the Darss peninsula is validated by comparing the water level time series generated by the model with local water level gauge data available at selected points (**Figure 33**). The Jupyter Notebook interface of the DT facilitates this comparison. Since the validation is against local data, it is to the user to provide the necessary validation data. In our use case, we retrieved data via www.pegelonline.wsv.de for the storm Babet in October 2023. The gauge at Barhoeft was also used for the boundary conditions at the entrance of the inlet. The locations Barth and Althagen are further inside the inlet and peak water levels show good agreement with the observed water levels. Flood extents are only compared to those generated by the flood monitor (dask-flood-mapper), guided qualitatively by news reports of dike breaches/overtoppings. Inclusion of dike breaches improves the agreement between the SFINCS and flood monitor floodmaps, see **Figure 34**. More in-depth comparison between the two floodmaps is limited by factors such as a different water mask (e.g. the reed rim along the water's edge), the timing of the satellite overpass relative to the peak water level, and the sensitivity of the flood monitor in coastal areas. The satellite passed over the area approximately one day after the peak water levels at the entrance of the inlet or half a day inside the inlet respectively (**Figure 33**). Flood waters may have receded already to some extent on the satellite images leading to a smaller area flooded when comparing to the maximum flood extents generated by the model.



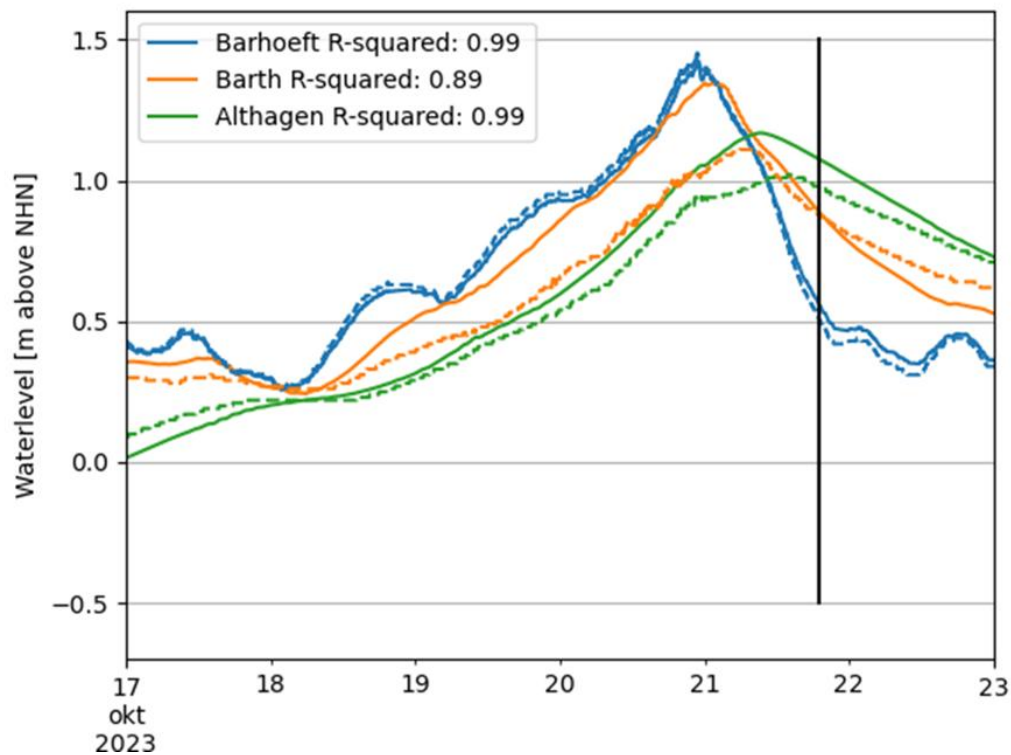


Figure 33 SFINCS validation by comparing waterlevels observed at the green triangles in Figure 6b) (dashed lines) with the SFINCS output (solid lines). The solid black line indicates the timing of the satellite overapss

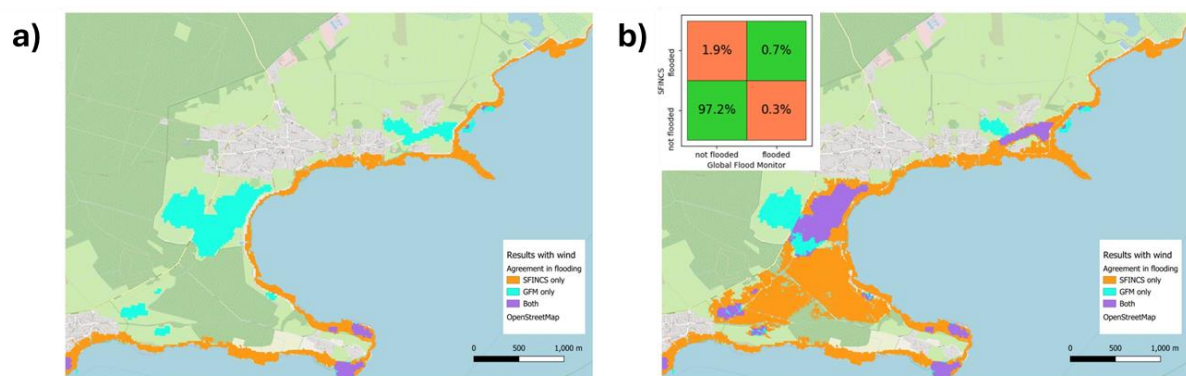


Figure 34 Agreement between Global Flood monitor and SFINCS for a small section of the Darss peninsula without (a) and with (b) dike breaches in the SFINCS model. The inset in b) shows the percentage agreement over the full domain

3.6 Alpine droughts early warning

3.6.1 DTs Application Integrations

The inputs of the wflow_sbm hydrological model are produced and processed from dynamical meteorological forcings and static parameter maps by running the HydroMT component. These inputs are open source datasets (Table 6) that are collected and registered into a Spatio-Temporal Assets Catalog.

Table 6 Input Parameters from Open Source Datasets

Collection/Dataset	Parameter	Source	STAC
EMO1	total_precipitation	https://data.jrc.ec.europa.eu/dataset/0bd84be4-cec8-4180-97a6-8b3adaac4d26	https://stac.intertwin.fedcloud.eu/collections/EMO1_TA24 PR RG PET DAILY
	orography		
	2m_temperature		
Jansen_Haise PET	Potential Evapotranspiration	Processed using PyET from EMO1	https://stac.intertwin.fedcloud.eu/collections/EMO1_TA24 PR RG PET DAILY
SoilGrid 2000	Bulk density	https://files.isric.org/soilgrids/latest/data/bdod	stac.eurac.edu:8080/collections/SOILGRIDS
	Organic carbon	https://files.isric.org/soilgrids/latest/data/soc	
	Clay	https://files.isric.org/soilgrids/latest/data/clay	
	Silt	https://files.isric.org/soilgrids/latest/data/silt	
	Sand	https://files.isric.org/soilgrids/latest/data/sand	
	pH	https://files.isric.org/soilgrids/latest/data/phh2o	
	Soil thickness	https://files.isric.org/soilgrids/former/2017-03-10/data/BDTICM_M_250m_IL.tif	
Corine Land Cover 2018	land cover class	http://s3.eu-central-1.amazonaws.com/eumap/lcv/lcv_landcover.hcl_lucas.corine.rfp_30m_0..0cm_2018_eumap_epsg3035_v0.1.tif	https://stac.eurac.edu/collections/CLC2018
Merit hydro	flow direction	https://hydro.iis.u-tokyo.ac.jp/~yamada/MERIT_Hydro/	stac.eurac.edu/collections/MERIT_HYDRO
	elevation		
	HAND		
	upstream area		
	river width		
HydroLakes	-	https://www.hydrosheds.org/products/hydrolakes	No
Grand v1.3	-	https://ln.sync.com/dl/bd47eb6b0/anxaiqr-62pmrgtq-k44xf84f-pyz4atkm/view/default/447819520013	No
Randolph Glaciers inventory	-	https://cds.climate.copernicus.eu/cdsapp#!/dataset/insitu-glaciers-extent?tab=overview	No



Roughness river mapping	-	https://github.com/Deltares/hydront_wflow/tree/main/hydromt_wflow/data/wflow	No
Corine mapping	-	https://github.com/Deltares/hydront_wflow/tree/main/hydromt_wflow/data/lulc	No
river_ge30m	-	https://zenodo.org/records/3552776#.YVbOrppByUk	No

The processed input-output of wflow_sbm is required for training the surrogate model (Table 7).

Table 7 Processed input-output of wflow_sbm

Collection/ Dataset	Parameter	Source	STAC
Wflow sbm forcings	Precipitation	Processed by HydroMT	Yes
	PET	Processed by HydroMT	Yes
	Temperature	Processed by HydroMT	Yes
Wflow sbm static maps	effective parameters⁴⁷	Processed by HydroMT	Yes
Wflow sbm outputs	Actual evapotranspiration (ET)		Yes
	SSM		Yes

The satellite-based SSM product from TU Wien was utilized for running the parameter learning (Table 8).

Table 8 Dataset to perform parameter learning

Collection/ Dataset	Parameter	Source	STAC
RT0	SSM	Processed by TU Wien	https://services.iodc.eu/browser/#/v1/collections/SSM-RT0-SIG0-R-EXTR

⁴⁷ https://deltares.github.io/Wflow.jl/stable/model_docs/params_vertical/

Workflow

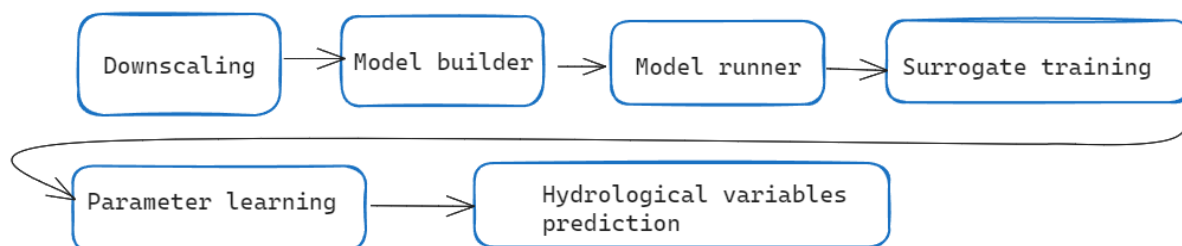


Figure 35 High-level diagram of the DT workflow's application components

Downscaling component (WP 7.4). DownscaleML

To streamline the downscaling process using downScaleML⁴⁸ version 0.9.0 (WP 7.4), we have developed preprocessing and downscaling methodologies that use reanalysis datasets as both predictors and references. This version introduces grid cell-wise statistical downscaling, employing ERA5 predictor fields and incorporating EMO1 data as targets for 2m temperature, precipitation and solar radiation. This approach utilises a two-stage downscaling approach involving recursive feature elimination, an improved hybrid LGBM classifier-regressor framework, random grid cell-based fine-tuning of hyperparameters, and is subsequently applied over SEAS5 to obtain downscaled SEAS5 outputs. This final version also makes extensive use of STAC and openeo-processes-dask seamlessly integrating with OSCAR and the openEO interTwin backend.

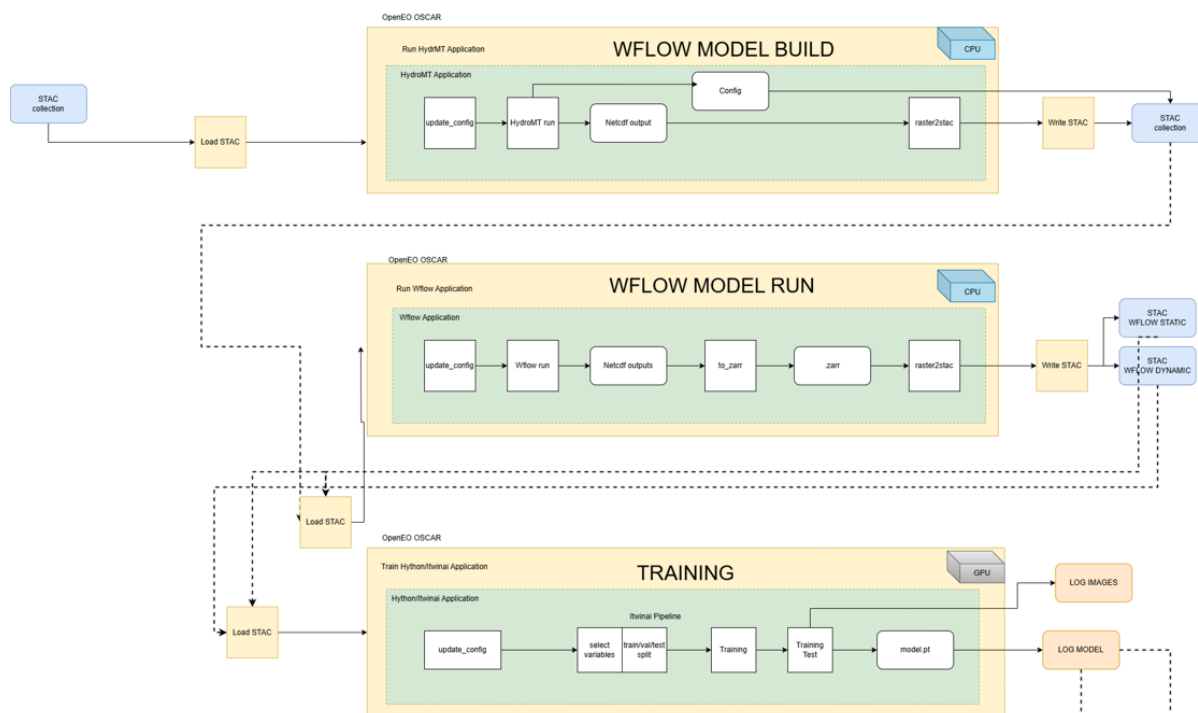


Figure 36 High level diagram that shows how DT's application components are run

⁴⁸ <https://github.com/interTwin-eu/downScaleML>

Model builder and model runner components (WP 7.6).

The model builder (HydroMT) and model runner (wflow_sbm) are available as Docker images in the HyDroForm⁴⁹ repository. The integration of HydroMT with STAC has been developed in a forked development branch⁵⁰ waiting to be tested and integrated officially in the version 1.0.0 of HydroMT. The implementation currently allows the generation of wflow_sbm model with different spatial resolutions and precipitation inputs available on the STAC catalog.

The openEO integration (WP 6.1, 6.3) consisted in the definition of a dedicated OpenEO Run_Oscar process that can trigger the execution of the docker images in OSCAR.

Surrogate training component.

The DL surrogate has been successfully developed and it is available in the project repository Hython⁵¹, where demo notebooks are available for the interested users. The integration with *itwinai* (WP 6.5) framework is completed for the preprocessing and training steps and consists in a dedicated itwinai-Hython plugin⁵².

The openEO integration followed the same approach of the model builder and model runner.

Parameter learning component.

The requirements for the production of SSM data to perform the parameter learning task have been shared with TU Wien. TU Wien then produced SSM over the Alpine region (WP 7.5).

This version uses a radiative transfer model to derive soil moisture data from 500m microwave backscatter and Leaf Area Index (LAI). To enhance the accuracy of the soil moisture retrievals, high-resolution information has been included from Sentinel-1 VV backscatter data (20 m). A novel soil moisture sensitivity dataset [R8] was used for a static spatial filtering technique at the 20m scale. Additionally, extreme backscatter values (higher than -5dB or lower than -19dB) have been filtered out, assuming that they are unlikely to contain reliable soil moisture information. Both strategies amplify the soil moisture signal of the 500m backscatter information, to which the 20m sub-pixels have been aggregated subsequently. Only those 500m target pixels were calculated that consisted of more than 1% valid 20m pixels after the described filter processes.

Based on this 500m backscatter datacube and LAI data, the radiative transfer model parameters have been calibrated using 4 years of data from 2016 up to 2020 with ERA5-Land swvl1 as reference data. During calibration, the frozen soil conditions indicated by ERA5-Land soil temperature layer 1 (stl1) and snow depth (sd) have been masked out to make sure only valid data is used during model inversion.

After calibration, the parameters were used to estimate SSM at a 500m resolution, without further masking to avoid losing valid data, especially over alpine valleys. This

⁴⁹ <https://github.com/interTwin-eu/HyDroForm>

⁵⁰ <https://github.com/iacopoff/hydromt>

⁵¹ <https://github.com/interTwin-eu/hython>

⁵² <https://github.com/interTwin-eu/hython-itwinai-plugin>



decision requires users to manually mask out data from frozen soils, and over steep and complex terrain. Despite its limitations in handling complex terrains, the initial version shows promising results, particularly in alpine valleys. By incorporating volume scattering using LAI and masking soil moisture insensitive pixels at high 20m resolution, we anticipate significant improvements in SSM retrievals compared to earlier models.

A validation of the product against available in-situ soil moisture time series was performed. If the product gaps are greatly reduced compared to current operational SSM products, the quality of the retrievals, as expected on complex terrains, was not satisfactory for an integration into the hydrological model workflow. This further motivates research in improving satellite retrievals in the Alpine region. The parameter learning task is fully supported in itwinai-Hython but was not integrated into openEO due to time limitations.

Seasonal forecast component.

It was not possible to integrate the inference step in Itwinai due to the fact that the framework is maturing and is currently not supporting running complex inference tasks. The inference can be executed in a Jupyter Notebook and does not require GPU devices.

3.6.2 Scope and limitations

The goal is to provide a user interface, consisting in the openEO GUI, that allows:

1. Set up the wflow_sbm model, selecting the model resolution, choosing different data inputs, time period and spatial domain.
2. Training the surrogate model.
3. Perform the parameter learning.
4. Run seasonal forecasts of hydrological variables.

The DT presents three main limitations:

- The DT predicts hydrological variables based on the type and level of detail of hydrological processes represented in the wflow_sbm model. This fact limits the variety of hydrological processes that could instead be better represented by running an ensemble of hydrological models.
- The Alpine region, with its complex terrain, vegetation cover and diverse climatic conditions, poses a great challenge for the retrieval of SSM and actual evapotranspiration from satellites. It is no surprise then that there are currently no products that satisfactorily cover the region. The products generated for the DT should therefore be considered as prototypes that require further research, development and validation. The parameter learning task's loss function is based on these satellite products and therefore depends on their quality.
- The DT does not represent human processes, such as irrigation and hydropower generation, that affects the hydrologic cycle.

3.6.3 Validation and Results

The main objective of the DT is to simplify the set up and running of complex workflows, linking model chains and heterogeneous data, for hydrological forecasting. The DT implements a novel hydrological model calibration approach that leverages a DL emulator and satellite products. A thorough validation of the calibrated model outputs requires observations of the target variables such as soil moisture, evapotranspiration and snow water equivalent, which are rather scarce in the Alpine region. The DT's seasonal forecasts have been validated against past events, for example the 2017 and 2022 droughts. In this case the performance of the DT is assessed against the hydrological model baseline (forced by historical observations) or even against the output of other seasonal forecast systems. The quality of the hydrological forecasts is very much dependent on the quality of the meteorological forecasts, which are too coarse to be useful on the Alpine region and are known to contain biases. For this reason the DT applies a downscaling and bias correction before running the hydrological model. Results show that the downscaling and bias correction can be improved as the system is not able to reproduce the 2022 drought on the Po valley **Figure 37**.

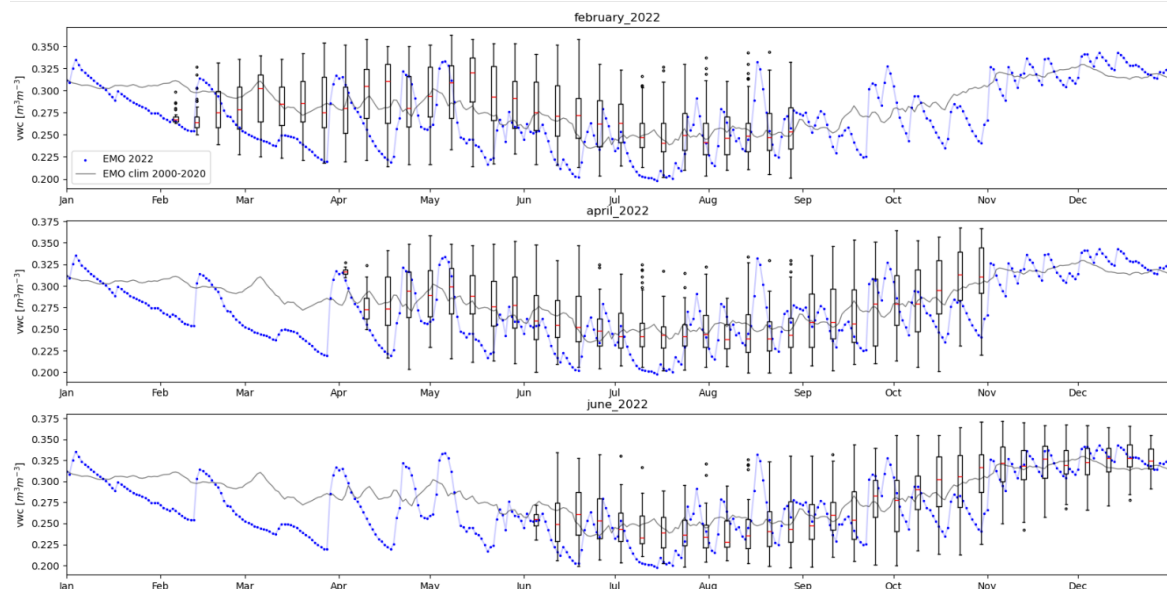


Figure 37 The image shows the volumetric water content (VWC, i.e. SSM) for three initialization dates, February, April and June, averaged over the Po basin. The blue dots represent the 2022 VWC, the grey line represent the 2000-2020 VWC climatology, and the boxplots represent the forecasted VWC. It is clear that the forecasts are rather following the climatology and can't really capture the 2022 drought

3.7 Flood climate impact in coastal and inland regions

3.7.1 DTs Application Integrations

Model requirements

The flood climate impact DT consists of two physics-based models, two impact models, and a python package facilitating processing the models:

1. SFINCS: a reduced-complexity model designed for super-fast modelling of compound flooding in a dynamic way.

2. Wflow: a framework for modelling hydrological processes, allowing users to account for precipitation, interception, snow accumulation and melt, evapotranspiration, soil water, surface water and groundwater recharge in a fully distributed environment.
3. Delft-FIAT: a fast, flexible, Python-based tool to rapidly assess direct economic impacts to buildings, utilities, and roads for user-input flood maps.
4. RA2CE: a Resilience Assessment and Action perspective for Critical infrastructure – model for mapping the exposure, criticality, and vulnerability as well as the forthcoming prioritisation of locations to take actions based on cost benefit assessment.
5. HydroMT: an open-source Python package that facilitates the process of building and analysing spatial geoscientific models with a focus on water system models. It does so by automating the workflow to go from raw data to a complete model instance which is ready to run and to analyse model results once the simulation has finished.

Workflow

The workflow for the flood climate impacts DT (**Figure 38**) is implemented in a Jupyter Notebook and is exploiting the following components from the project:

- WP7: The workflow depends on functionalities from the hydrological model data processing thematic module dependent on SFINCS⁵³, Wflow⁵⁴, Delft-FIAT, and RA2CE and their respective HydroMT plugins for the first three models to generate flood and impact maps.
- WP6: CWL workflows have been developed integrating SFINCS, Wflow, Delft-FIAT, RA2CE, and HydroMT to generate flood and impact maps. The model runs are triggered using the WP6 component OSCAR, with the option to offload the job to HPC should the model run time require increased computational resources.
- WP5: Data for model building and preprocessing is accessed from the interTwin Data Lake using Rucio. The OSCAR jobs to run the various models can either be triggered through OSCAR directly, or through Rucio events by registering the required input data in the Data Lake. An activity is in progress to investigate triggering the OSCAR jobs directly from the Data Lake instead of using Rucio events.

⁵³ <https://github.com/Deltares/SFINCS>

⁵⁴ <https://github.com/Deltares/Wflow.jl>



D4.7 Final version of the DTs capabilities for climate change and impact decision support tools including validation reports

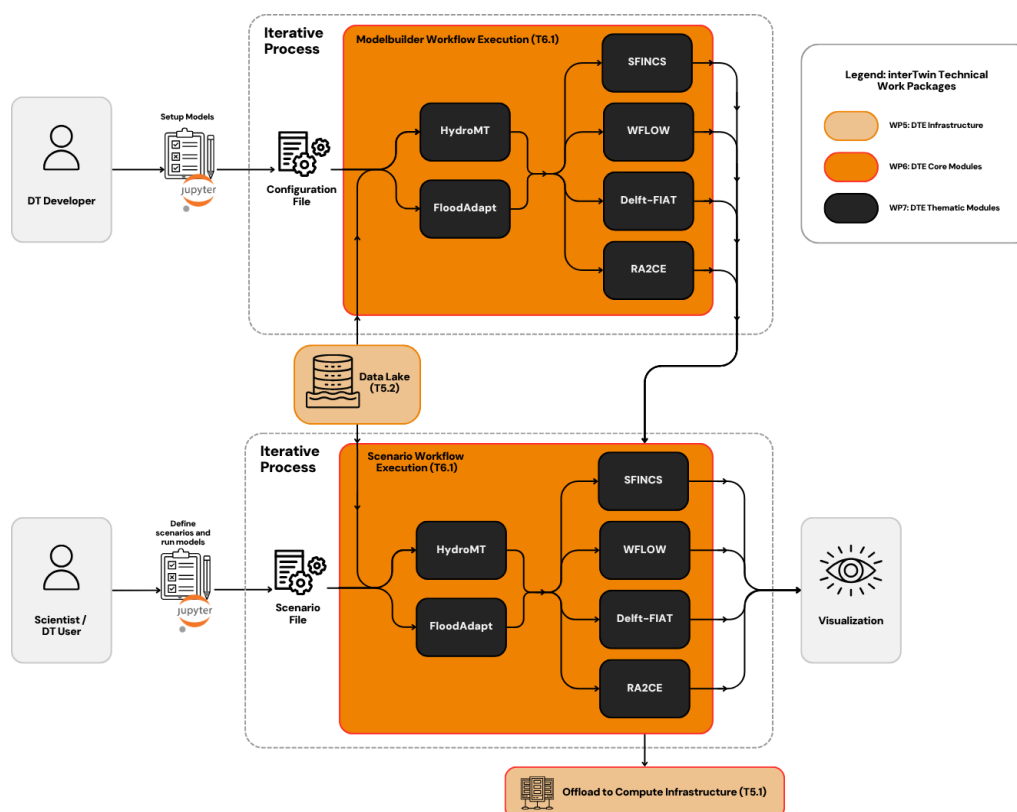


Figure 38 Architecture diagram highlighting integration and interaction between the Jupyter Notebook (the DT interface), the CWL workflows and interTwin's RUCIO-based datalake

3.7.2 Data requirements

The datasets currently used by the DT are listed in **Table 9**. These include both static parameters maps for SFINCS, WFLOW, Delft-FIAT, RA2CE and dynamic meteorological forcing data for SFINCS, WFLOW.

Table 9 Datasets used by the DT

Dataset Name	Source	Parameter Name
ERA5	https://doi.org/10.24381/cds.bd0915c6	2m Temperature
		Mean Sea Level Pressure
		10m Wind u-component
		10m Wind v-component
		Total Precipitation
		Surface net Solar Radiation
Copernicus 30m DEM	https://spacedata.copernicus.eu/documents/20123/121286/Copernicus+DEM+Open+HTTPS+Access.pdf	Topography

GEBCO	https://www.gebco.net/	Bathymetry
GTSM reanalysis	https://doi.org/10.24381/cds.8c59054f	Waterlevel
		Surge
Hydro Atlas	https://www.hydrosheds.org/hydroatlas	Basin, lake, & reservoir atlas
ESA worldcover	https://doi.org/10.5281/zenodo.5571936	Landclass
GCN250 Infiltration curves	https://doi.org/10.6084/m9.figshare.7756202.v1	Average antecedent
		Dry Antecedent
		Wet Antecedent
MERIT_HYDRO	http://hydro.iis.u-tokyo.ac.jp/~yamada/ME_RIT_Hydro	flwdir
		strord
		uparea
River_ge30m	https://zenodo.org/records/3552776#.YVbOrppByUk	River width, bankfull discharge
MODIS Leaf Area Index	https://lpdaac.usgs.gov/products/mcd15a3hv006/	lai
Soilgrids	https://www.isric.org/explore/soilgrids/faq-soilgrids-2017	soilmaps
OpenStreetMap	https://osmnx.readthedocs.io/en/stable/	Building footprints
GADM	https://gadm.org/data.html	level 3, level 4 administrative boundaries
Worldpop	https://hub.worldpop.org/doi/10.5258/SOTON/WP00684	Population count

3.7.3 Scope and limitations

This use case provides Jupyter Notebooks for expert users in order to

1. Set up the necessary models for a user-defined region of interest.
2. Run the necessary models to produce baseline flood maps for a user-defined region of interest and quantify impacts and damages to buildings, utilities, roads and accessibility.
3. Select flood mitigation and adaptation measures and re-run flood scenarios to test their effectiveness at reducing flood-related impacts.

A demonstration is provided for Humber, UK.



3.7.4 Preconditions

The user has access to DT data, models, thematic components and Jupyter Notebooks.

Users can:

- Specify a region of interest,
- Specify a temporal period to simulate,
- Select and specify mitigation and adaptation measures.

The user runs the DT workflows for the specified region and scenario using default global data or selected local data if available.

The output of the DT can be visualised in the Jupyter Notebooks and the data can be downloaded/saved as NetCDF and GeoPackage data.

3.7.5 Validation and Results

The primary objective of this DT is to help users set up initial versions of the models used in the workflow using global data and run what-if / hypothetical scenarios based on past events. Validating such a flood model derived from global data against local data would immediately indicate that the first improvements to be made are using local data sources. Making such an improvement is part of the Flood Early Warning DT, which shares thematic components with this DT. See section 3.4.4 for details.

Nevertheless, running what-if scenarios can still be meaningful with this DT when comparing against baseline events run using the same model chain, particularly regarding insights on the effect of adaptation measures. The DT provides users with interactive visuals to inspect the impact of scenarios run with the DT, examples of which are shown in **Figure 11** and **Figure 12**.

4 Conclusions

The final release of the interTwin DTs applications for WP4 concerned with the environmental domain were developed during the whole duration of the project.

The DT applications on wildfires and tropical cyclones are supported by multiple components from the DTE (e.g., RUCIO and interLink from WP5, Ophidia, yProv and itwinai from WP6, and a few thematic components from WP7). For both applications different CMIP6 datasets have been made available on the interTwin Data Lake for simplifying the execution of the application. The Ophidia capabilities have been used either for preprocessing climate datasets or running the inference pipelines on multiple climate data. In terms of results, the UNet++ architecture selected for the implementation of the wildfires DT application showed good skills in capturing the global patterns of wildfires. For the TC application, different approaches for detection and tracking have been explored with the final version being able to capture most of the tracks and follow them closely. For both DT applications, notebooks are provided for enabling user interaction with the DT application, while simple configuration files are available for more expert users for training new ML models.

The Eddies detection application demonstrated how interTwin technologies can be easily exploited to enable, with a limited effort, the porting and development of new DTs. Besides that, from a performance perspective, the DT-based version of the application significantly outperformed the data-intensive one (by 3 orders of magnitude).

The DT application on the detection and impacts of generic extreme events is supported by components developed in both WP6 and WP7. This application has been pre-configured for users to detect heatwaves, but it can also be tailored by the DT users to other climate extremes and climate variables, thanks to its use of an architecture based on CVAE, which is a generic anomaly detection method based on AI. The use of *itwinai* from WP6 made the integration of the DT into the interTwin DTE easy thanks to the plug-in capabilities of *itwinai*. Like the DT applications on wildfires and tropical cyclones, it uses data from the CMIP6 database, and notebooks are provided to users to interact with the DT application. A configuration file is provided to users on how to properly configure the application to target heatwaves. The notebook that is provided aims to show users how to interact with the application, and how to post-process data and display results.

The DT application on drought early warnings on the Alpine region demonstrated that complex, reproducible workflows that integrate heterogeneous cloud-based data with process-based and DL models can be customized and executed leveraging the InterTwin technologies. The use of DL emulators to calibrate the process-based model, which was possible thanks to the *itwinAI-Hython* plugin, showed promising results, nonetheless the integration of satellite retrievals still requires further improvements. On the other hand, the performance of hydrological seasonal forecasts to reproduce drought events will require further research and development, especially in the Alpine region where complex topography and climate heterogeneity pose a great challenge to the predictability power of forecasts.



The DT applications on Flood Early warning and on Flood climate impact demonstrate the capabilities of the DTE in supporting the impact assessment and future planning for extreme flood events. The DT Flood early warning shows how process-based inundation models can be enhanced by satellite-derived floodmaps, by guiding and fine tuning the setup process of the physics-based flood model. The notebook and underlying integrated workflow that guides a user through this process is supported by WP5 and WP6 for data and compute infrastructure management, on which the thematic components from WP7 are built. The utility of the DT is demonstrated by the use-case along the German Baltic coast where the satellite-derived floodmap provides additional information on dike breaches that help fine-tune the inundation model.

The DT Flood climate impact allows the user to get a first estimate of the impact of extreme flood events under current and future climate and socio-economic conditions. The notebook interfaces first guide the user through the process of setting up the necessary hazard and impact models based on global data sources, supported by the data management from WP5. Next, using the thematic components from WP7 the user can specify the event they would like to assess and under which climate and socio-economic conditions they would like to do so. The assessment is run using an integrated workflow running and coupling the various hazard and impact models together, supported by WP6 for the management of compute infrastructure. A use case demonstrating the DT capabilities was developed for the Humber estuary in the UK.

5 References

Reference	
No	Description / Link
R1	An ensemble machine learning approach for tropical cyclone localization and tracking from ERA5 reanalysis data. G. Accarino et al. In Earth and Space Science 2023, vol 10, e2023EA003106 DOI: 10.1029/2023EA003106
R2	Very deep convolutional networks for large-scale image recognition. K. Simonyan and A. Zisserman. arXiv preprint, 2014 DOI: arXiv:1409.1556
R3	SEAS5: The new ECMWF seasonal forecast system. Johnson, S. J., Stockdale, T. N., Ferranti, L., Balmaseda, M. A., Molteni, F., Magnusson, L., Tietsche, S., Decremmer, D., Weisheimer, A., Balsamo, G., Keeley, S. P. E., Mogensen, K., Zuo, H., & Monge-Sanz, B. M. (2019). <i>Geoscientific Model Development</i> , 12(3), 1087–1117. DOI: 10.5194/gmd-12-1087-2019
R4	Integrating Scientific Knowledge with Machine Learning for Engineering and Environmental Systems. Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2022). <i>ACM Computing Surveys</i> , 55(4). DOI: 10.1145/3514228
R5	Differentiable modeling to unify machine learning and physical models and advance Geosciences. Shen, C., Appling, A. P., Gentine, P., Bandai, T., Gupta, H., Tartakovsky, A., Baity-Jesi, M., Fenicia, F., Kifer, D., Li, L., Liu, X., Ren, W., Zheng, Y., Harman, C. J., Clark, M., Farthing, M., Feng, D., Kumar, P., Aboelyazeed, D., ... Lawson, K. (2023). DOI: 10.1038/s43017-023-00450-9
R6	Continuous streamflow prediction in ungauged basins: Long Short-Term memory neural networks clearly outperform traditional hydrological models. Arsenault, R., Martel, J. L., Brunet, F., Brissette, F., & Mai, J. (2023). <i>Hydrology and Earth System Sciences</i> , 27(1), 139–157. DOI: 10.5194/hess-27-139-2023
R7	Satellite-Based Flood Mapping Through Bayesian Inference from a Sentinel-1 SAR Datacube. Bauer-Marschallinger, Bernhard, Senmao Cao, Mark Edwin Tupas, Florian Roth, Claudio Navacchi, Thomas Melzer, Vahid Freeman, and Wolfgang Wagner. (2022). <i>Remote Sensing</i> , 14 (15): 3673. DOI: 10.3390/rs14153673
R8	Enabling global scale Sentinel-1 time series analysis through streaming. Raml, B., Vreugdenhil, M., Massart, S. J. A., Navacchi, C., & Wagner, W. (2023). Enabling global scale Sentinel-1 time series analysis through streaming. In P. Soille, S. Lumnitz, & S. Albani (Eds.), <i>Proceedings of the 2023 conference on Big Data from Space (BiDS'23): From foresight to impact</i> (pp. 29–32). Publications Office of the European Union. DOI: 10.34726/5308



R9	Overview of the Coupled Model Intercomparison Project Phase 6 (CMIP6) experimental design and organization. Eyring, V., Bony, S., Meehl, G. A., Senior, C. A., Stevens, B., Stouffer, R. J., and Taylor, K. E., Geosci. Model Dev., 9, 1937–1958, 2016. DOI: 10.5194/gmd-9-1937-2016 .
R10	High Resolution Model Intercomparison Project (HighResMIP v1.0) for CMIP6, Geosci. Haarsma, R. J., Roberts, M. J., Vidale, P. L., Senior, C. A., Bellucci, A., Bao, Q., Chang, P., Corti, S., Fučkar, N. S., Guemas, V., von Hardenberg, J., Hazeleger, W., Kodama, C., Koenigk, T., Leung, L. R., Lu, J., Luo, J.-J., Mao, J., Mizielinski, M. S., Mizuta, R., Nobre, P., Satoh, M., Scoccimarro, E., Semmler, T., Small, J., and von Storch, J.-S., Model Dev., 9, 4185–4208. DOI: 10.5194/gmd-9-4185-2016 , 2016.
R11	The Scenario Model Intercomparison Project (ScenarioMIP) for CMIP6. O'Neill, B. C., Tebaldi, C., van Vuuren, D. P., Eyring, V., Friedlingstein, P., Hurtt, G., Knutti, R., Kriegler, E., Lamarque, J.-F., Lowe, J., Meehl, G. A., Moss, R., Riahi, K., and Sanderson, B. M., Geosci. Model Dev., 9, 3461–3482, 2016. DOI: 10.5194/gmd-9-3461-2016
R12	interTwin D4.5 Final Architecture design of the DTs capabilities for climate change and impact decision support tools. Elia, D., Donno, D., Donno, E., Ferrario, I., Fronza, M., Backeberg, B., Tromp, W., & Pagé, C. (2025). DOI: https://doi.org/10.5281/zenodo.15096734