Ref. Ares(2025)5304277 - 02/07/2025



Scalable Parallel Astrophysical Codes for Exascale

SPACE ML framework #1 Release

Deliverable number: D3.3

Version 1.0/1.0





Funded by the European Union. This work has received funding from the European High Performance Computing Joint Undertaking (JU) and Belgium, Czech Republic, France, Germany, Greece, Italy, Norway, and Spain under grant agreement No 101093441

Project Information

Project Acronym: SPACE
Project Full Title: Scalable Parallel Astrophysical Codes for Exascale
Call: Horizon-EuroHPC-JU-2021-COE-01
Grant Number: 101093441
Project URL: https://space-coe.eu

Document Information

Editor:	Sebastian Trujillo Gomez, HITS
Deliverable nature:	Report (R)
Dissemination level:	Public (PU)
Contractual Delivery Date:	30.06.2025
Actual Delivery Date	30.06.2025
Number of pages:	33
Keywords:	Data analytics, machine learning, artificial intelligence, workflows
Authors:	Guillermo Marin – BSC Eva Sciacca – INAF Sebastian Trujillo Gomez – HITS Bernd Doser – HITS Andreas Fehlner – HITS Kai Polsterer – HITS Petr Strakos – IT4I@VSB Iacopo Colonnelli – UNITO Doriana Medić – UNITO Yolanda Becerra – BSC Enric Sosa – BSC Dario Garcia – BSC Marta Barroso – BSC Pablo Martin – BSC Arnau Miró – BSC David Ferrer – BSC
Peer review:	Robert Wissing – UiO Luca Tornatore – INAF

History of Changes

Release	Date	Author, Organization	Description of changes
0.1	16.01.2025	Eva Sciacca, INAF	Provided template
0.2	6.3.2025	Sebastian Trujillo G.	Added document structure
0.9	17.6.2025	all	Final edits
0.9.5	24.06.2025	Robert Wissing, UiO; Luca Tornatore, INAF	Internal review
1.0	30.6.2025	all	Final document



DISCLAIMER

Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European High Performance Computing Joint Undertaking (JU) and Belgium, Czech Republic, France, Germany, Greece, Italy, Norway, and Spain. Neither the European Union nor the granting authority can be held responsible for them.

The space above and below the message intentionally is left blank.

Executive Summary

This deliverable, D3.3 "SPACE ML framework First Release", is a report on the efforts in Task 3.3 towards developing the Machine Learning (ML) tools proposed in the project proposal.

The development of our ML tools and solutions is guided by the work reported in D3.1 "SPACE ML, visualisation data analysis and workflow framework use cases and requirements", where we identified three use cases for ML, along with their respective user requirements. The main criteria for the current selection was: i) the relevance for the Astrophysics and Cosmology (A&C) community, and ii) that they represented challenges in the field of simulation development and data analysis that are common to other scientific domains. A more thorough description of the Use Cases, requirements, and the methods used to gather them is available in D3.1.

In the following sections we discuss the first release of results for the three Machine Learning Use Cases:

1. Representation Learning for Explorative Knowledge Discovery in Simulations
2. Physics-informed reconstruction using synthetic observations of cosmological simulations
3. Predicting the effect of radiation on gas cooling rates on-the-fly in hydrodynamical simulations

Contents

1 Introduction **6**

2 Use Cases **7**
2.1 Representation Learning for Explorative Knowledge Discovery in Simulations 7
2.1.1 Introduction 7
2.1.2 Tool development 7
2.1.3 Work plan for release #2 11
2.2 Physics-informed reconstruction using synthetic observations of cosmological simulations 14
2.2.1 Introduction 14
2.2.2 Tool development 14
2.2.3 Model Training 18
2.2.4 Evaluation 21
2.2.5 Results 21
2.2.6 Limitations 22
2.2.7 Work plan for release #2 22
2.3 Predicting the effect of radiation on gas cooling rates on-the-fly in hydrodynamical simulations . 24
2.3.1 Introduction 24
2.3.2 Tool development 25
2.3.3 Results 26
2.3.4 Work plan for release #2 27

3 Conclusions **29**

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

List of Figures

1 Architecture of the convolutional Hyperspherical Variational Autoencoder (HVAE) model implemented at the core of `Spherinator`. . . . 8
2 Output of the `HiPSter` explorative visualization module. *Left:* Interpretation of the representation space learned by `Spherinator`. The simulated galaxy images are generated by the decoder of a trained HVAE. *Right:* Embedded dataset represented at the highest (i.e. most general) level of the Hierarchical Progressive Survey (HiPS) hierarchy by the images of the galaxies nearest to the center of each tile. . . . 10
3 Schematic workflow for training and inference using our tools for explorative access to large cosmological simulations. . . . 11
4 Configuration of the workflow orchestration cluster using the Flyte cloud-native platform. . . . 11
5 Schematic of our new concept for a complete discovery workflow using exascale cosmological simulations. . . . 12
6 High level overview of the pipeline . . . 15
7 Velocity histograms of the main halo (black), Intra Cluster Light (ICL) (red), and Brightest Cluster Galaxy (BCG) (blue) from Dianoga (1). Black dotted line corresponds to fitting a Single Maxwellian distribution to the velocities of all particles. The Double Maxwellian fit is represented with a dashed black line. . . . 16
8 Transition radius and cumulative mass of ICL, BCG for halo=0 in TNG300-1 . . . 17
9 Example of ICL mass fraction (first row) and velocity dispersion (second row) maps for halos = $\{0, 1001, 1002, 1003\}$ and projection = $\{x, y\}$ in TNG300-1 . . . 19
10 Adapting State Space Models to work with 2D sequences: the input is patched and traversed in four directions to produce sequences which are then fed to a SSM block. Results are aggregated at the end. Opacity is proportional to the amount of transformations applied to the hidden state after first seeing that patch. . . . 20
11 Overview of the functioning of a 2D *selective SSM* block (SS2D). In addition to the sequence processing units (SSM), which parametrize equations (1) and (2), the block contains linear + convolutional projections, as well as residual connections, which function on sequences of patches coming from each transversal direction. In our architecture, SS2D blocks are concatenated one after the other in an encoder-decoder structure. After every two blocks, feature dimensionality is reduced via patch merging in the encoder, and augmented via patch expanding in the decoder to allow for hierarchical representation learning of VD features. . . . 20
12 (Left) Scaling laws computed for all architectures on 3091 halos from the Illustris and Eagles dataset. Mamba scales comparatively better than other models. (Right) ICL predictions using a Mamba architecture. . . . 23
13 On both plots: normalized, absolute, net cooling rates for the Solar component ($\Lambda_{i,\odot}$) for redshift zero. The left plot shows the cooling rates as a function of density and temperature while on the right shows a horizontal cut of the left plot, i.e., the cooling rates as a function of the temperature at a fixed density. . . . 25
14 Flowchart diagram of the ML model workflow. . . . 26
15 Multi-Layer Perceptron (MLP) results on the metal-free group (left) and Solar group (right). . . . 27
16 Training loss for the metal-free group (left) and Solar group (right). . . . 27
17 Scaled data for the metal-free group (left) and Solar group (right). . . . 28
18 MLP results on the metal-free group (bottom) and Solar group (top). Left: original data. Right: results of the MLP evaluation. . . . 29

List of Tables

1 Overview of selected simulations used in this work, including the simulation code, box length in comoving Mpc/h, number of halos selected and filtered for analysis, number of generated Velocity Dispersion and Intra Cluster Light maps and the estimated transition radius (in physical kpc) used to separate the BCG and ICL regions. . . . 17

2 Model evaluation on the Dianoga dataset (lower is better). Metrics are Relative Average Spectral Error (RASE), Robust Mean Squared Error (RMSE). ICL Frac corresponds to ICL fraction absolute difference between ground truths and predictions, or Mean Absolute Error. TFLOPs are computed per forward pass on a single sample. 22
3 Model evaluation on the unseen Dianoga dataset of models trained on a huge dataset containing only halos from TNG300 and Eagle. RASE-FT shows the performance obtained after a single fine-tuning epoch on 6 halos from the Dianoga dataset. 22
4 Model evaluation on the unseen Dianoga dataset of models trained on a huge dataset containing only halos from TNG300 and Eagle. RASE (1 epch) shows the performance obtained after a single fine-tuning epoch on 6 halos from the Dianoga dataset. 22

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

List of Acronyms

A&C	Astrophysics and Cosmology
AI	Artificial Intelligence
CFD	Computational Fluid Dynamics
CoE	Centre of Excellence
HPC	High Performance Computing
CNN	Convolutional Neural Network
HVAE	Hyperspherical Variational Autoencoder
HiPS	Hierarchical Progressive Survey
PEST	Preprocessing Engine for Spherinator Training
SBI	Simulation Based Inference
MLP	Multi-Layer Perceptron
ICL	Intra Cluster Light
BCG	Brightest Cluster Galaxy
ML	Machine Learning
SPACE	Scalable Parallel Astrophysical Codes for Exascale

1 Introduction

This deliverable D3.3 "SPACE ML framework #1 Release" constitutes the initial report on the development and deployment of the ML framework envisioned within the Scalable Parallel Astrophysical Codes for Exascale (SPACE) Centre of Excellence (CoE). Building upon the identified use cases and requirements presented in deliverable D3.1, this document provides an overview of the advancements made in creating scalable and efficient ML tools specifically designed for the A&C community. The ML tools developed under SPACE Task 3.4 "Machine learning for astrophysics" aim to address critical challenges related to subgrid modelling, data exploration and analytics, and interpretation of predictions obtained from cosmological simulations, particularly within the context of Exascale computing.

The foundational basis for this release is the comprehensive identification of ML use cases and the corresponding user requirements documented in D3.1, which outlined crucial application scenarios including knowledge discovery, simulation-based reconstruction, and surrogate subgrid models in cosmological simulations. Deliverable D3.3 presents significant progress made in these key areas corresponding to the three identified use cases:

- Representation Learning for Explorative Knowledge Discovery in Simulations;
- Physics-informed reconstruction using synthetic observations from cosmological simulations;
- Real-time prediction of radiation effects on gas cooling rates within hydrodynamical simulations.

The document details the methodologies, initial results, evaluations, and development workflows for each use case, emphasizing modularity, scalability, and adherence to FAIR principles for scientific data and codes (i.e. Findability, Accessibility, Interoperability, and Reusability). Furthermore, it outlines the planned enhancements and additional functionalities to be incorporated in subsequent releases toward the end of the project. This deliverable thus not only serves as a milestone marking the successful first-stage deployment of these ML solutions but also as a stepping stone towards more sophisticated and comprehensive tools tailored for research based on Exascale A&C simulations.

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

2 Use Cases

We describe the design and development of the tools for the three identified use cases for Machine Learning methods below.

2.1 Representation Learning for Explorative Knowledge Discovery in Simulations

Lead WP3 partner: HITS

2.1.1 Introduction

Our solution consists of a self-contained tool that takes the raw simulation data as input, performs all the preprocessing under the hood, and trains a model to learn a compact interpretable representation of user-selected structures (such as galaxies) in arbitrary physical components. For this, we avoid representing galaxies using summary statistics (e.g. scalars including luminosity, colors, effective radii, Sérsic index, star formation rate, etc.) or 1D representations along arbitrary projections (e.g. surface brightness profiles, mass profiles, etc.) guided only by human intuition. Instead, the tool approaches the complete data agnostically, using Generative Deep Learning methods to learn the most efficient representation of simulated cosmological structures (i.e. galaxies) in a low-dimensional latent space. The latent space dimensionality is guided by the requirement that it should be easily visualized, explored, and interpreted by humans.

This guarantees that the data can be inspected easily and interactively, even for the largest cosmological simulations of the exascale era. The galaxies in this reduced representation space can then be mapped to with the traditional summary statistics to aid interpretation and enable the discovery of hidden relationships. They may also be colored and annotated using latent representations of the extrinsic variables (like formation history or environment) to find and investigate the causal drivers of observed galaxy properties. The visualization is lightweight such that it can run on any laptop via a web server, and it provides the functionality to interactively inspect and select subsamples of data for local analysis.

2.1.2 Tool development

Cosmological hydrodynamical simulations are excellent numerical laboratories to understand astrophysical processes like the formation of galaxies and large-scale structure. Richard Feynman's famous quote, "What I cannot create, I do not understand" summarizes why these numerical laboratories are essential to understand the Universe we observe. Cosmological simulations provide a highly detailed realization of simulated universes across a vast range of spatial and temporal scales, from shortly after the Big Bang until the present day. Each simulation snapshot is rich in high-dimensional data, including the 6D phase space coordinates for each matter component (e.g. dark matter, gas, stars, black holes), as well as dozens of physical properties for all components, e.g. density, temperature, metallicity, individual element abundances, etc.).

State-of-the-art large-volume cosmological simulations represent the evolution of the Universe using more than 10^{11} particles and produce petabytes of data products. The size and complexity of the data already far exceed the exploration, synthesis, and interpretation capacity of humans, with the rise of the exascale era rendering traditional analysis techniques obsolete. Moreover, architectures and I/O formats vary greatly across simulation codes, hindering the application of software tools generally across many different cosmological simulation datasets. Simulation data is represented in a highly compressed format in catalogs, often collapsing the rich multidimensional data to single scalars (e.g. galaxy and halo properties). By utilizing dimensionality reduction techniques, we strive for automatically learning a more meaningful compressed representation of the original data. Machine learning provides powerful tools to compress data to an arbitrarily low number of dimensions, while ensuring that similar objects have similar compressed representations. The latent representation can be used to automatically create searchable similarity catalogs of simulated galaxies or halos that encode much more information than traditional halo catalogs. This is done by capturing the most relevant information at the particle level (in 3D point clouds) or pixel level (in 2D images), automatically including complex morphological galaxy features as well as halo shapes and mass distributions. This is also, in essence, what historical empirical relations between galaxy properties encode, including the Hubble sequence, the galaxy main sequence, the mass-metallicity relation, the Tully-Fisher relation, and the fundamental plane, among others.

As many tools in astronomy exist to visualize and handle data on a sphere, we decided to evaluate the use of HiPS (2) to represent an ML-derived projection of simulation data on a spherical surface. The hierarchical nature

of HiPS allows the user to iteratively refine the view, providing explorative access to the large, complex, and highly-structured simulation outputs. Our tools provide an essential contribution to simulation data findability and accessibility in the context of the FAIR data principles (3). The dimensionality reduction is handled by the `Spherinator` module, while the HiPS representation is created by the `HiPSter` module.

`Spherinator`: To achieve a meaningful embedding in a lower-dimensional space, neural networks have proven to be powerful ML approaches. In recent years, the HVAE has emerged as a powerful tool in the field of generative modelling (4; 5), utilizing a spherical surface for the embedding. Due to its novel geometry, the hyperspherical latent (i.e. compressed) space enables a more efficient and meaningful exploration of the compressed representation. Unlike the traditional Euclidean space, the hyperspherical space is periodic and constrained in size. The first version of `Spherinator` uses PyTorch Lightning to implement a variational autoencoder (VAE) with a 2D spherical latent space and using a Convolutional Neural Network (CNN) for both the encoder and decoder.

CNN layers are designed to extract progressively abstract features from images by convolving them with a set of filters that are learned from the data during training. We implemented an CNN encoder network using a five-layer cascade of convolutional and pooling layers that progressively reduce the image size while increasing the number of channels, i.e. with dimensions $[3 \times 128^2, 16 \times 64^2, 32 \times 32^2, 64 \times 16^2, 128 \times 8^2, 256 \times 4^2]$ (where the three channels of the input correspond to RGB filters). This is followed by a fully connected layer with output dimension [256], and two independent fully connected layers that together produce a 3D feature vector. The feature vector represents a density distribution on a spherical surface using a simple renormalized 3D vector of unit length for the 2D coordinates of the mean μ, and a rotationally symmetric concentration parameter κ. A fully connected layer then takes the 2D sample from this distribution and converts it into a vector of dimension [256] that becomes the input of the decoder network. The decoder is simply the inverted version of the encoder. The bottleneck between the encoder and the decoder forces the network to learn an efficient compression of each image from its native pixel space, $\mathbb{R}^{3\times128\times128}$, onto a feature vector in $\mathbb{R}^3$. A sketch of the network architecture is shown in Figure 1.

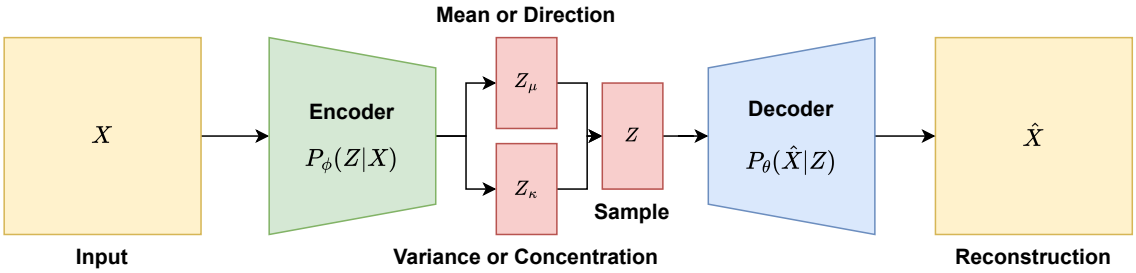


Figure 1: Architecture of the convolutional HVAE model implemented at the core of `Spherinator`.

To enforce rotational invariance of the learned representations of galaxy images, the reconstruction loss is calculated for a set of rotated images and their reconstructions, and only the version with the lowest loss is used for optimizing the weights and biases of the network. The loss function consists of two parts: the image reconstruction loss, and the Kullback-Leibler (KL) divergence,

$$L = L_{recon} + \lambda \cdot L_{KL}.$$

The reconstruction loss is measured using the pixel-wise root-mean-square deviation (RMSE), while the KL divergence measures the deviation of the encoder $P_\phi(Z|X)$ from the desired latent distribution $P_\theta(Z)$. The KL divergence effectively acts as a regularizer to prevent overfitting, and ensures a well-behaved latent space (6; 7). A Gaussian distribution is assumed for Euclidean latent spaces, while a uniform distribution is used for hyperspherical latent spaces. The contributions to the loss from the reconstruction and the KL divergence are balanced using the factor λ, which is adjusted to ensure a meaningful latent space (8).
The power-spherical distribution is simply a normal distribution on the surface of a hypersphere, defined as

$$p_X(x; \mu, \kappa) = N_X(\kappa, d)^{-1}(1 + \mu^\top x)^\kappa,$$

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

where μ is the direction, κ is the concentration, and N_X is the normalization factor. Since the KL divergence between a uniform distribution and a power spherical distribution is independent of the direction μ, the regularization term can be expressed directly using κ. We use the Adam optimizer with a learning rate scheduler from 10^{-3} to 10^{-5} and batch size 256 to train the model for 200 epochs. The data was augmented using 36 rotated versions of every input image (i.e. a full 360 deg rotation in increments of 10 deg). We found $\lambda = 10^{-3}$ to be a good choice to produce a reasonable representation of galaxy morphology.

`HiPSter`: As soon as a model is trained, the `HiPSter` is used to create an explorable representation of the projected data-set. Hereby, we utilize the HiPS standard to store the results in a hierarchical form. This enables us to represent extremely large data-sets in a very compact way, a first step to the data we expect from exascale simulations. To generate all necessary files, the following tasks had been implemented.
Model Representation: Based on the trained autoencoder, we utilize only the generative decoder part to produce an image for the center coordinates of each HEALpix cell. Those images are stored following the HiPS standard alongside with a properties file.
Catalog Creation: The encoder part is used to embed all input images on the sphere. This task is done, using the same technique for rotation invariance as described for the training of the model. The resulting coordinates from the bottleneck as well as some meta-information are written to a CSV file and a VOTable[1].
Image Embedding: Based on the created catalog file, for each order of the HiPS, the tiles are filled with the input image that projects closest to the center of the corresponding HEALpix cell. In case no input image maps to a cell, the corresponding tile is left empty.
Thumbnails & Originals: We generate JPEG images for a hover on presentation and to allow inspecting the original uncropped and unscaled input images within AladinLite.
All-sky Images: To support Aladin Sky Atlas on the desktop, overview images are produced for every order following the HiPS standard.
Deployment: To deploy the produced two HiPS tilings as well as the catalog files, an HTML file is automatically created. This file utilizes AladinLite to provide direct visual explorative access to the representation learned with the `Spherinator` as presented in a web demo at `https://space.h-its.org`.

Figure 2 shows the output of `HiPSter` for the largest publicly available dataset of synthetic galaxy images ($\sim 6 \times 10^4$ galaxies from the Illustris and IllustrisTNG simulation suites[2] (9; 10; 11).

Interactive Usage: When the tiles and catalogs are access via Aladin Sky Atlas on the desktop, users can annotate regions of interest, create own catalogs by using multi-order coverage maps, search by similarity by querying sources in the vicinity through e.g. a cone search, or identify outliers by looking at those sources with the largest reconstruction errors.

Applications: A whole new explorative data access workflow arises from the representation of morphological structures as coordinates on a sphere, enabling the realization of very diverse scientific use cases including

- Model-driven experiment design: identification of testable novel predictions
- Bayesian Optimization of subgrid models
- Model selection through statistical comparison with observational survey data
- Simulation based inference for cosmological parameters and galaxy formation physics

Adaptive Scalable Representation Learning Workflows: Cosmological simulations typically output an enormous amount of data, with the largest ones exceeding $\sim$ 1 PB. Enhanced techniques and methods are required to process data of this size and complexity efficiently. Workflow orchestration is of critical importance to modern data-driven astrophysics in several respects as it enables

- reproducibility of computations by tracking the code, data, models, and environments,
- efficient compute resource management by running tasks in parallel and optimizing resource usage,
- pipelines to be scaled up to handle large amounts of data and complex pipelines,

[1] `https://www.ivoa.net/documents/VOTable/`
[2] `https://www.tng-project.org/`

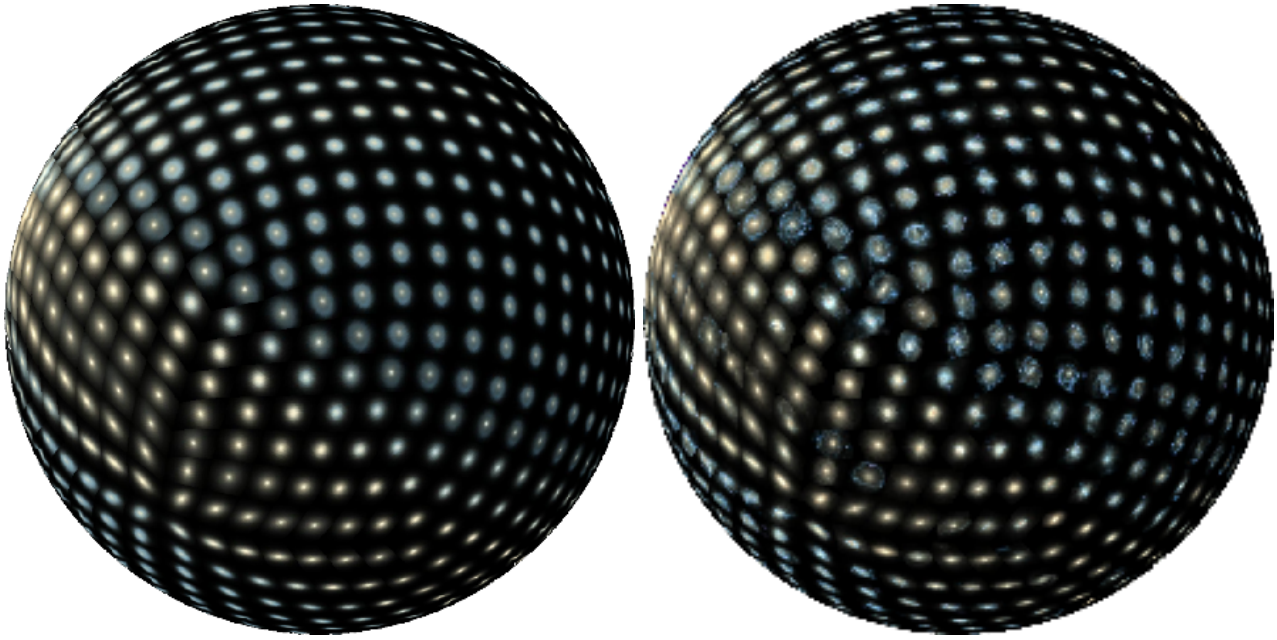


Figure 2: Output of the `HiPSter` explorative visualization module. *Left:* Interpretation of the representation space learned by `Spherinator`. The simulated galaxy images are generated by the decoder of a trained HVAE. *Right:* Embedded dataset represented at the highest (i.e. most general) level of the HiPS hierarchy by the images of the galaxies nearest to the center of each tile.

- tracking workflow progress and monitoring performance and results, and
- sharing code, data, and results to improve collaboration.

Here we present a comprehensive workflow to facilitate integrating current and emerging dimensionality reduction techniques (12). This workflow takes the FAIR principles into account, ensuring that these foundational guidelines are appropriately addressed. We present two distinct but complementary workflow orchestration frameworks: one specifically designed for High Performance Computing (HPC) clusters and one optimized for cloud-native environments.

The workflow (see Figure 3) is designed with an adaptable structure, featuring general modules for data preprocessing, model training, prediction (model inference), and deployment, making it suitable for various tasks. As a first step, the Preprocessing Engine for Spherinator Training (PEST) is designed to process many types of data, including astrophysical simulations and observations, into abstract universal representations regardless of their origin. PEST converts raw data into a unique dataset optimized for machine learning model training and inference. The module uses Apache Parquet as a data format optimized for reading and writing data frames due to its columnar storage structure. The model training can use a deep learning-based variational autoencoder (Spherinator), self-organizing Kohonen maps (PINK), or any other algorithm that may become available in the future, e.g. involving Zernike polynomials or graph neural networks (GNNs). The trained models are exported into the ONNX (Open Neural Network Exchange) format, an open standard for compatibility with various frameworks and programming languages. The prediction module performs the inference using the trained model. For example, the HiPSter module projects the spherical latent space of the Spherinator module onto HiPS tiles. The final deployment module performs the user interaction using Aladin Lite to visualize the hierarchical spherical latent space, and Jasmine to display 3D structures interactively.

Workflow Orchestration: Workflow orchestration consists of the definition and execution of workflows and data pipelines on a cloud-native or HPC cluster. However, since the operational methods of HPC clusters (typically managed by Slurm) and cloud-native clusters (typically managed by Kubernetes) differ fundamentally, we have employed two separate orchestration frameworks adapted for each case.

StreamFlow[3] is available as a Python package and has a simple installation process that makes workflow

[3] https://github.com/alpha-unito/streamflow

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

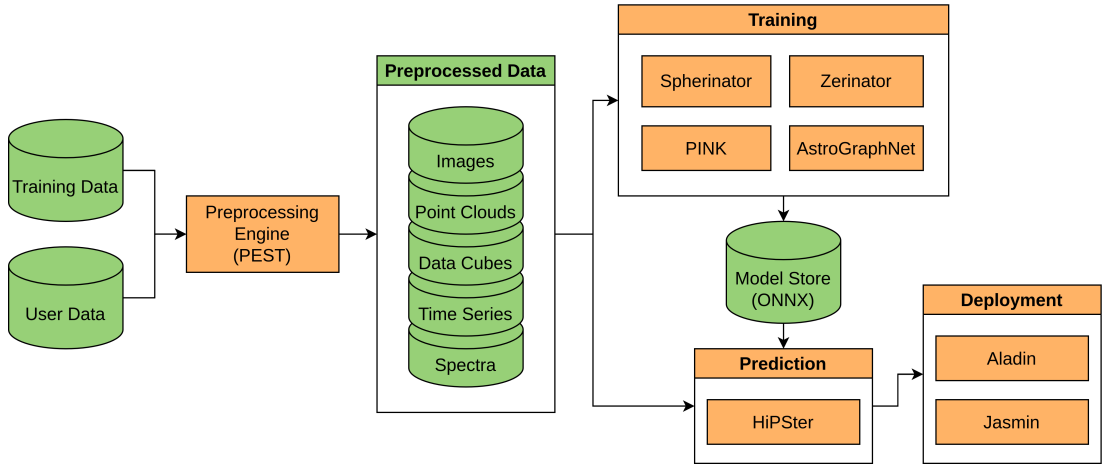


Figure 3: Schematic workflow for training and inference using our tools for explorative access to large cosmological simulations.

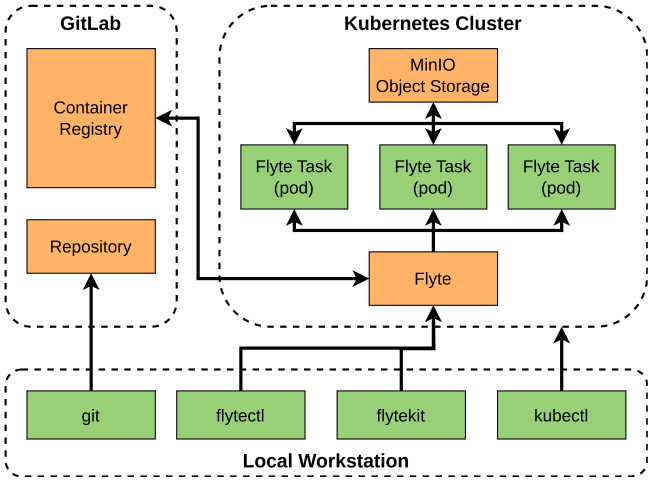


Figure 4: Configuration of the workflow orchestration cluster using the Flyte cloud-native platform.

distribution easy. The relationships between tasks are defined using the Common Workflow Language (CWL), and data is mainly exchanged through a distributed file system. StreamFlow interacts seamlessly with the Slurm scheduler, providing an ideal solution for orchestration on HPC clusters, where the compute nodes can only be used via Slurm.

Flyte[4] is a highly scalable, cloud-native workflow orchestration platform that leverages containers, Kubernetes, and object storage to facilitate data pipelines. Figure 4 provides a schematic overview of the Flyte architecture. The Flyte platform, which encompasses the administration interface, scheduler, and dashboard, is integrated within a Kubernetes cluster. Workflows are registered through a command line interface provided by Flyte. Each task within the workflow is executed in a dedicated Kubernetes pod and container, and data is transferred between tasks via object storage.

The requirements of Flyte to the environment are more complex than those of StreamFlow, but this also allows for a more flexible approach to defining workflows. Flyte's data pipeline utilizes Python's static type system and object storage, offering significantly more stability than the StreamFlow solution.

2.1.3 Work plan for release #2

With the goal of enabling new and efficient scientific discovery workflows based on exascale cosmological simulations, we have identified a more general set of requirements for our exploration and interpretation framework

[4] https://github.com/flyteorg/flyte

that extend well beyond our original objectives. We aim to design our framework to provide a single flexible interface that can integrate all available and future Representation Learning architectures under a single user-friendly open-source interface, using a modular structure that can be extended and updated by the community to avoid obsolescence and lift the burden of constantly learning new interfaces from the user. Out complete discovery framework for exascale cosmological simulations will consist of universal modular tools that:

- operate on abstract data structures,
- automatically create compact lossless representations of multimodal data without labels,
- enable exploration and interpretation of arbitrarily large datasets using interactive hierarchical visualization,
- allow direct comparison with next-generation galaxy surveys (for inference and model selection).

Figure 5 shows a schematic overview of the architecture of our complete discovery framework, highlighting the three main components of the workflow: data pre-processing, data-driven modeling, and discovery.

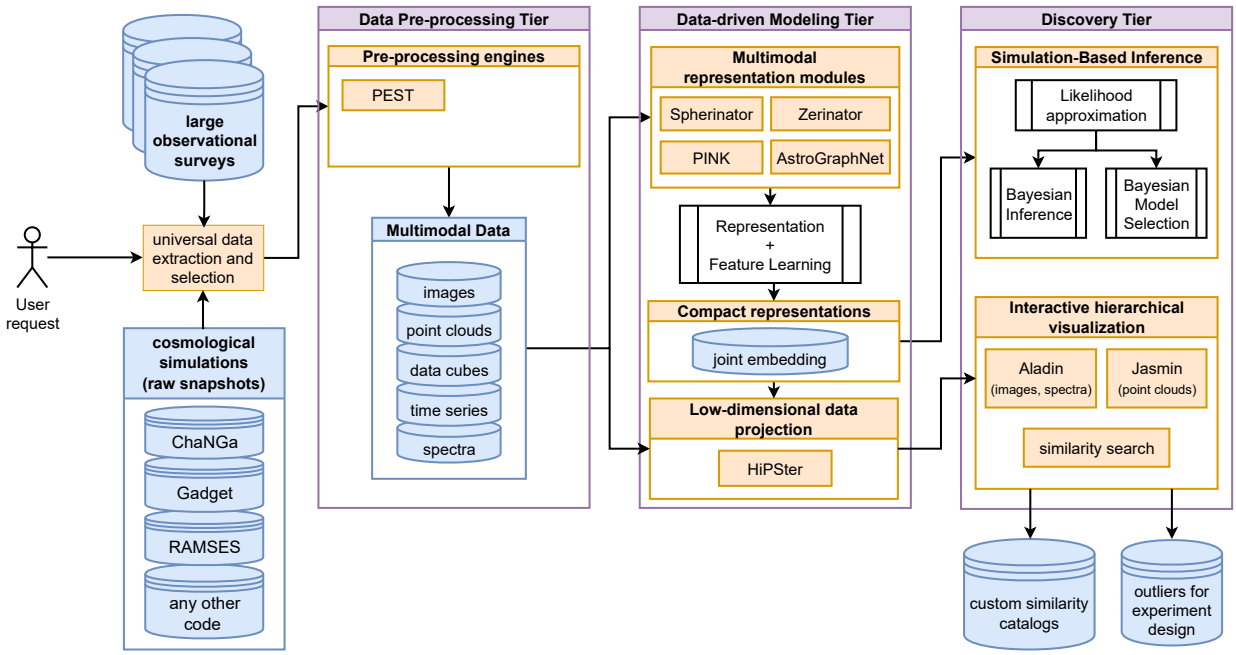


Figure 5: Schematic of our new concept for a complete discovery workflow using exascale cosmological simulations.

In the remaining 18 months of SPACE we will focus mainly on extending PEST to 1) transform the raw simulation outputs (particle data and halo catalogs) into abstract representations using the Apache Parquet ML-optimized open-source data format, and then 2) produce custom user-requested representations of the galaxies (including summary statistics, 2D/3D maps, synthetic optical spectra, images or spectral datacubes, HI position-velocity datacubes, 3D/6D point clouds, etc.) to feed into the multimodal representation modules in the data-driven modeling stage. We plan to include modular data readers provided by each of the cosmological code partners to couple the individual codes to PEST, as well as a framework for the community to provide data readers for any other simulation code.

If time permits, we also plan to integrate additional multimodal representation modules that are currently under development: `Zerinator` for learning rotationally invariant representations of 2D maps and images using Zernike polynomials, `AstroGraphNet` for learning 3D/6D point cloud representations of simulated galaxies using Graph Neural Networks, and `PINK` (13) for learning rotation and flipping-invariant Kohonen maps.

Lastly, a discovery framework for exascale simulations would be incomplete without a Bayesian inference engine to obtain posteriors on cosmological and galaxy formation physics through robust statistical comparison with the wealth of observational data that will soon be released by next-generation observatories including

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

Euclid, Rubin, and the Square Kilometer Array (SKA). Once the data pre-processing stage is ready, we plan to complement the discovery stage using recently developed and robust techniques for Simulation Based Inference (SBI), also known as Implicit Likelihood Inference. For this we can take advantage of the learned compressed representation which is essential to enable SBI techniques based on neural network approximations to the unknown likelihood. The compressed representation can be fed to a state-of-the-art modular, flexible, and extensible SBI framework such as LtU-ILI[5] (14). For this purpose we plan to develop a module to convert the compressed representation learned in the modeling tier into the corresponding format. By including a similarity search module, the discovery tier will allow observers to easily query any cosmological simulation for galaxies that are structurally similar to observed objects of interest, including for example very high redshift disks or local ultra diffuse galaxies (UDGs). Although most of the modules mentioned above were not part of the initial plan, we expect to be able to complete these extra developments by utilizing additional in kind resources.

[5] https://github.com/maho3/ltu-ili

2.2 Physics-informed reconstruction using synthetic observations of cosmological simulations

Lead WP3 partner: BSC

2.2.1 Introduction

Due to the complexity of the physical processes involved in cosmological galaxy formation and evolution, sophisticated numerical simulations are often necessary to interpret observational data. This is usually done in a hand-wavy way by comparing the simulated and observed galaxies using summary statistics (e.g. luminosity, morphology, colors, stellar mass, star formation rate, effective radius, etc.), restricting then to the simulated objects that match the observations to obtain the desired physical properties. This traditional approach has many limitations, including the reliance on summary statistics, and the introduction of systematics when comparing inferred quantities such as stellar mass. Moreover, it lacks a rigorous and robust statistical framework and does not produce uncertainty estimates.

To address these challenges, we propose a general-purpose pipeline designed to infer the physical properties of galaxies by training a variety of state-of-the-art deep learning models on synthetic observed maps obtained from cosmological simulations. To illustrate the flexibility and potential benefits of this tool, we apply it to a real scientific use case: using integral field spectrograph (IFU) data of galaxy clusters to map the 2D spatial distribution of stars that are not bound to galaxies, i.e. the ICL. This general framework can then easily be extended to infer many other physical property maps such as stellar mass, star formation rate, age, metallicity, assembly history, dark matter fraction, etc.

The intracluster light, composed of stars unbound from galaxies, is a diffuse stellar component that traces the dynamical evolution of galaxy clusters. Its spatial distribution and kinematic properties offer valuable insights into past interactions, such as mergers and tidal stripping (15), which drive the hierarchical assembly of structure in the universe. However, detecting and characterizing the ICL remains a complex task, as it typically exhibits low surface brightness and overlaps significantly with the BCG (16). In both simulations and observations, its definition varies, leading to methodological inconsistencies and limiting direct comparisons.

Here we propose a general-purpose pipeline designed to generate realistic synthetic observations from state-of-the-art cosmological hydrodynamical simulations. We then use deep learning to learn a mapping between the observables (velocity dispersion maps) and physical properties (the map of the ICL fraction). The goal is to create 2D maps with multiple features that reflect both the physical properties of clusters and the spatial geometry of modern integral field spectrographs (IFUs) such as MUSE [6]. These maps combine structural and dynamical information, including velocity dispersion and ICL fraction, and are intended to serve as inputs for machine learning models capable of identifying and separating diffuse components under observational constraints. The high-level overview of the pipeline, as well as examples of maps, is shown in Figure 6.

This section presents the structure and implementation of the pipeline, which is designed to support several simulation suites (ensuring compliance with the FAIR data principles) and offers configurable control over image generation, feature pairing, spatial resolution, and observational realism. Particular care is taken in capturing overlapping structures and simulating projection effects, noise, and point spread functions. The resulting framework enables the systematic creation of synthetic datasets that are well suited for developing and evaluating ICL classification models, ultimately contributing to more consistent and physically motivated analyses of the diffuse stellar content in clusters.

All experiments, including data preprocessing, map generation, and model training, were conducted on the MareNostrum 5 (MN5) supercomputer at the Barcelona Supercomputing Center (BSC). The pipeline is designed to be modular and scalable, making it compatible with HPC environments. The ability to run on HPC resources such as MN5 ensures reproducibility, scalability, and efficient handling of the computational demands required by high-resolution cosmological simulations and deep learning models.

2.2.2 Tool development

The construction of our pipeline is based on a modular and interoperable architecture, designed to transform raw simulation data into structured 2D maps for machine learning applications. Each core stage - data loading, rotation, and map generation - is implemented as an independent module, coordinated through a flexible

[6] MUSE (Multi Unit Spectroscopic Explorer) is an instrument on the Very Large Telescope (VLT) that captures both images and spectra across a wide field of view, allowing detailed spatial and spectral analysis of astronomical objects.

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

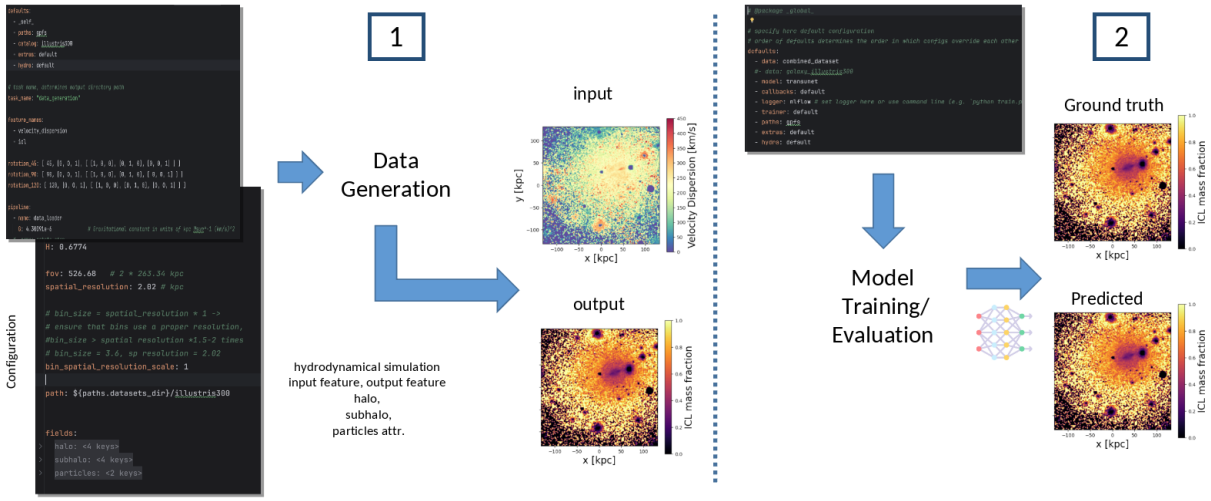


Figure 6: High level overview of the pipeline

execution controller. Simulation-specific configurations, such as file paths, naming conventions, and physical constants, are defined externally, allowing the core logic to remain general and reusable across datasets.

A key objective of the design is to ensure scalability and long-term extensibility, i.e. guaranteeing reusability in the context of the FAIR principles. As new simulation datasets become available or as scientific priorities shift, the pipeline can be extended with minimal effort. New features (e.g., metallicity, gas temperature, or stellar ages), new simulation formats, or even new processing stages (e.g., synthetic photometry or redshift slicing) can be added by registering lightweight plug-in modules. This allows researchers to tailor the pipeline to different scientific goals without rewriting or duplicating code. It also ensures scalability to the increasingly larger datasets expected from exascale simulations and next-generation observatories.

To manage the heterogeneity of input formats and physical models in simulations, the pipeline uses a configuration-driven architecture combined with object-oriented programming. A general configuration file defines the pipeline steps, constants, number of dimensions, zoom levels, and rotation parameters (angles and axes). Each simulation also has a dedicated configuration file that specifies file paths, available features (halos, subhalos, particles), unit conversions, filters, spatial resolution, and transition radius. All configurations are written in YAML and managed through the Hydra[7] framework, allowing structured overrides and clear reproducibility.

Internally, the data model follows an object-oriented structure in Python. Cosmological entities are represented as modular classes: `HaloContext` for sets of halos, `SubhaloContext`, `SubhaloParticles` and individual `Halo`,`Subhalo`, and `Particle` objects. This design creates a clean abstraction layer between raw data and processing steps. Once a simulation is integrated via its configuration and interface module, it can access the full set of downstream operations without further modification. This ensures flexibility, consistency, and ease of extension across diverse datasets.

Data Loading: The mock observations are generated from a diverse suite of state-of-the-art cosmological hydrodynamical simulations, designed to trace the formation and evolution of galaxy groups and clusters under different physical prescriptions. The following simulation sets have been employed in this work: TNG300 (17) (18) (19) (20) (21), Dianoga (22), Magneticum (23), Flamingo (24), Millennium-TNG (25), and Eagle (26).

From these simulations, we extract halos, subhalos and stellar particle data. More precisely, the list of attributes extracted are: halo (coordinates, M200, R200, id of primary subhalo), subhalo (Halo Id, Stellar mass, Coordinates, Stellar half-mass radius), stellar particles (coordinates, velocity, coordinates).

As part of the data loading process, these data is parsed and standardized using proper physical units. We apply a selection filter to retain halos with $M_{200} > 10^{13}\ M_{\odot}$ and subhalos with stellar mass $M_{*} > 10^{10}\ M_{\odot}$, consistent with thresholds used in previous work (1). This mass selection is crucial because it filters out underdeveloped systems, allowing us to focus on those with mature structures that contain substantial ICL,

[7]Hydra is a Python framework for managing hierarchical configurations using YAML files, allowing flexible and composable parameter control for complex workflows

which is essential for the analysis and separation of ICL features in the data.

For each halo and its linked subhalos, we retrieve their stellar particles. To ensure correct spatial relationships across periodic simulation boundaries, we apply a periodic boundary condition (PBC) correction to particle and subhalo positions. This ensures that relative distances between objects near the simulation box edges are accurately computed, avoiding artifacts that could arise from coordinate discontinuities in periodic volumes.

The separation between ICL and BCG is conducted given the velocity and position of the halo and its particles, leveraging the full 6D phase-space information available in hydrodynamical simulations. This approach is rooted in the assumption that ICL and BCG stars follow distinct kinematic distributions, which can be disentangled through dynamical modeling. Following the methodology of (1), we adopt a strategy inspired by `ICL-Subfind` (27), where the stellar velocity distribution in the central subhalo is modeled as a superposition of two Maxwellian [8] components — The assumption is that ICL stars (more unbound, hotter) and BCG stars (more bound, colder) follow two distinct Maxwellian distributions, which allows their separation based on velocity. Fig 7 shows the resulting velocity histograms, main halo star particles velocity distribution (in black) and the single BCG (blue) and ICL (red), belonging to one of the clusters in Dianoga.

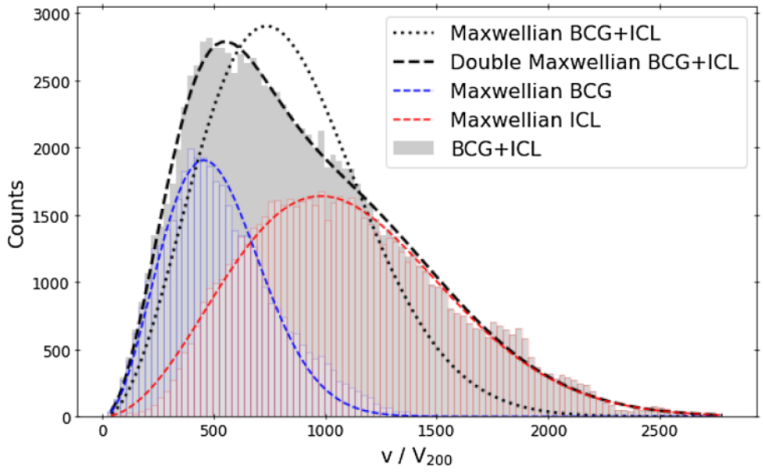


Figure 7: Velocity histograms of the main halo (black), ICL (red), and BCG (blue) from Dianoga (1). Black dotted line corresponds to fitting a Single Maxwellian distribution to the velocities of all particles. The Double Maxwellian fit is represented with a dashed black line.

Although the original `ICL-Subfind` algorithm iteratively adjusts this radius and fits the velocity distribution to perform the separation, its high computational cost makes it impractical for large datasets. Instead, we rely on the Random Forest classifier trained by Ilaria Marini et al. (1) on the output of `ICL-Subfind`, which uses features such as stellar velocity, host halo mass, and normalized particle distance to predict the ICL–BCG membership. This model, trained on the Dianoga simulation suite, provides an efficient and scalable proxy for dynamical separation. In our work, we extend its use to other cosmological simulations (TNG, EAGLE, Magneticum, among others), and validate its predictions by comparing the resulting ICL maps with physically motivated expectations, ensuring consistency across a range of formation histories and resolutions.

Before generating the final feature maps, we conducted an analysis to estimate the transition radius for each simulation - defined as the radial distance at which the ICL component becomes dominant over the BCG stellar contribution. This estimate is computed based on cumulative ICL, BCG and stellar mass profiles and is later used to define the spatial extent of ICL features in 2D maps and interloper masks. An example of a transition radius is shown in Figure 8.

To account for potential contamination from nearby structures, which can occur due to the line-of-sight projection in observational data, we incorporate an overlapping halo strategy. This strategy involves constructing a `cKDTree` [9], a spatial data structure, from the positions of all subhalos within a simulation snapshot. For each halo, we identify neighboring subhalos within a 3D sphere with a radius of R_{200}, where R_{200} represents the radius within which the halo's mean density is 200 times the critical density of the universe. Stellar particles

[8] A Maxwellian distribution (or Maxwell–Boltzmann distribution) describes the probability distribution of speeds or velocities of particles in a gas that is in thermal equilibrium.

[9] cKDTree is a binary tree that allows to quickly find the closest points to a given location in multi-dimensional data

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

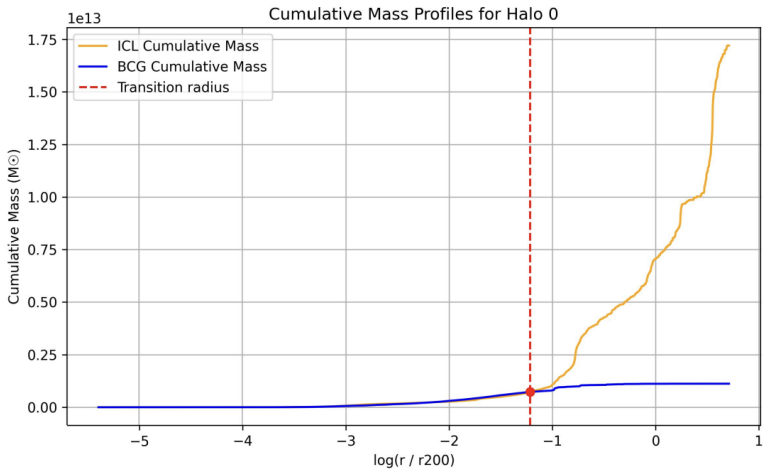


Figure 8: Transition radius and cumulative mass of ICL, BCG for halo=0 in TNG300-1

from these neighboring subhalos are included when feature maps are generated, reflecting the observational contamination that occurs in real-world imaging due to overlapping structures along the line of sight. This step is especially important in crowded cluster environments, where physical and projected structures often coexist closely.

To support model generalization and enable comparative studies across simulations, we build a unified dataset by aggregating data from the three out of six simulations mentioned above. These simulations span a wide range of resolutions, and formation histories, ensuring that the dataset encompasses a variety of dynamical environments and ICL morphologies. A summary of the number of halos, selected subhalos, and total synthetic images generated per simulation is provided in Table 1.

Simulation	Model	Box Lenght [Mpc/h]	# of Halos	# of Maps	TR [kpc]
Dianoga	-	1000.0	8	6241	263.3
TNG300-1	Fiducial	205.0	3032	658531	637.0
Eagle	AGNdT9L0050N0752	73.8	20	4537	277.1
Eagle	RefL0050N0752	73.8	20	4537	501.1

Table 1: Overview of selected simulations used in this work, including the simulation code, box length in comoving Mpc/h, number of halos selected and filtered for analysis, number of generated Velocity Dispersion and Intra Cluster Light maps and the estimated transition radius (in physical kpc) used to separate the BCG and ICL regions.

Bringing together data from these simulations into a single, standardized dataset represents a meaningful contribution to the community. It enables consistent training and evaluation of machine learning models across different cosmological contexts, helps identify how simulation-specific assumptions affect predictive performance, and lays the groundwork for training a foundational, generalizable model for ICL prediction.

Rotation: After the data is loaded, the particle positions and velocities undergo a rotation step. A random vector $\mathbf{z}'$ is generated such that the angle between the original z-axis and $\mathbf{z}'$ is fixed to 45°, introducing a consistent yet diverse orientation across samples. The axis and angle of rotation are configurable, allowing for controlled adjustments or full 3D randomization. This transformation serves as a form of data augmentation, increasing the variety of morphological and kinematic patterns encountered during training and contributing to improved generalization of the models.

Map Generation: After data preparation and rotation, we generate 2D maps for each halo, representing selected features such as ICL and velocity dispersion. The maps are constructed with a configurable spatial resolution and a fixed pixel size, which allows for flexibility in the representation of the data. Two essential parameters are set at this stage: the spatial resolution, which defines the number of bins or pixels in the final image, and the transition radius, which marks the radial boundary where the ICL begins to dominate over

the BCG. As mentioned previously, the transition radius is chosen based on a prior calibration analysis for each simulation, ensuring that the interface between the BCG and ICL is captured accurately within the map boundaries.

To align with realistic observational setups, the geometry of the generated maps is designed to mimic the Multi-Unit Spectroscopic Explorer (MUSE) Integral Field Unit (IFU) at the Very Large Telescope (VLT). The field of view is set to 1 arcmin^2 with a pixel resolution of 0.2×0.2 arcsec^2. This configuration is crucial as it allows for consistency in how the data is processed, ensuring that all systems are modeled within the same observational framework, even though only the spatial properties are considered in the maps.

In line with previous work by Marini et al. (2024) (28), we introduce random zoom levels during map generation. Specifically, images are slightly shifted by random offsets within 15% of the image size, adding spatial variability to the generated maps. This helps simulate the natural variability observed in real data. Additionally, to introduce further realism during data augmentation, we apply transformations such as adding random Gaussian noise. This noise simulates detector imperfections, which are common in real imaging systems. Furthermore, a point spread function (PSF) is simulated by applying a Gaussian blur, which mimics the spatial resolution degradation often seen in real astronomical observations. These transformations are applied only during the training process, ensuring that they do not alter the raw map outputs but enhance the variability during model training.

Maps are generated by projecting the particle data along a chosen line of sight, with per-pixel averages of the selected features calculated. For the ICL maps, the fraction of intracluster stars is computed within each spatial bin using a binary classification mask (1 = ICL, 0 = not ICL). Similarly, velocity dispersion maps are created by calculating the standard deviation of the projected velocities within each pixel, capturing the differences in dynamical heating across regions of the halo. These two features—ICL mass fraction and velocity dispersion—provide complementary views of the diffuse light and its kinematic state, offering critical information for distinguishing between different components within the system.

To address contamination from interloping particles—those belonging to nearby, overlapping halos but projected within the image plane—we flag these particles and store them in a dedicated binary mask channel within the output. This allows for later differentiation between in-situ ICL or BCG light and contamination from unrelated foreground or background structures. As highlighted in the work by Marini et al. (2021) (29), distinguishing between these components is crucial, particularly when comparing methods that treat such particles differently, such as dynamical versus photometric techniques.

Following these data transformations, the final dataset consists of synthetic images ready for training and analysis. Figure 9 provides example images of the ICL mass fraction (first row) and velocity dispersion (second row). Although a subtle correlation can be observed between the ICL mass fraction and the velocity dispersion maps, it is not immediately apparent to the human eye. However, certain features, especially near the BCG and extending outward, hint at the underlying relationship. This interplay between kinematic profiles of the BCG and ICL is expected to play a significant role in how deep learning models can identify and learn these patterns within the simulated images. To explore this, we employ a variety of model architectures, including convolutional models such as U-Net (30) and Attention U-Net (31), transformer-based models like TransUNet (32), and sequence modeling approaches based on Mamba (33). These architectures allow us to assess how different model types capture the spatial and statistical structure of the maps.

2.2.3 Model Training

Previous work on ICL prediction via Deep Learning models (Marini et al. (2024) (28) trained a U-Net architecture on mock images generated from simulation data. The U-Net (30) is a highly successful CNN architecture which consists on two components, an encoder which converts data to a lower dimensional space via convolutional kernels and pixel merging, and a decoder which converts the lower dimensional vectors output by the encoder into the target high dimensional data, ICL fraction maps in this case. Residual connections between corresponding encoder and decoder layers help preserve spatial information and improve gradient flow during training.

A variant of the U-Net is also introduced in (Marini et al. (2024) (28), the Attention U-Net (31), which incorporates cross attention gates on the residual connections to overcome existing limitations of the UNet architecture, such as the inability to effectively suppress irrelevant background features and a limited capacity to focus on relevant regions within the input data. Both are trained with a Mean Squared Error objective on mock images generated from the 80 most massive halos in the Dianoga simulation.

Our intuition is that long range dependencies play a crucial role in ICL prediction, given that we expect ICL

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

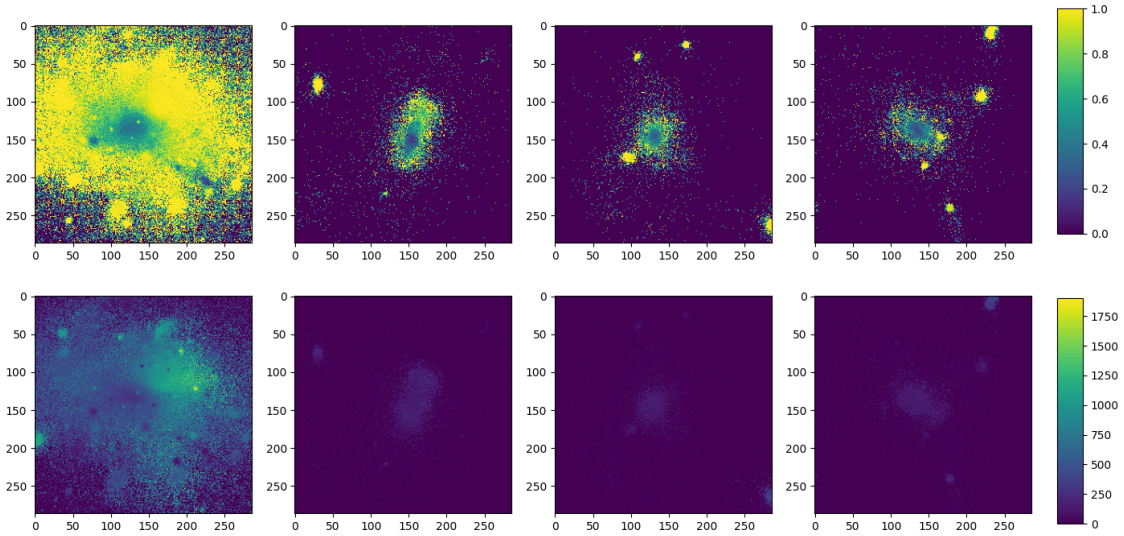


Figure 9: Example of ICL mass fraction (first row) and velocity dispersion (second row) maps for halos = $\{0, 1001, 1002, 1003\}$ and projection = $\{x, y\}$ in TNG300-1

and BCG particles to have different (global) velocity distributions. CNNs are limited in this respect, given that they consist on layers of local kernels. In this context, we explore TransUNet (32), a hybrid architecture that combines the strengths of CNNs and Vision Transformers. TransUNet replaces part of the U-Net encoder with a Transformer block that processes patchified image tokens, allowing the model to learn global representations while still preserving local spatial features through convolutional layers. However, attention mechanisms come with the trade-off of high computational cost (scaling quadratically with input length), which requires input images to be heavily patched to remain tractable. This issue becomes more pronounced as dataset sizes are increased, and poses a challenge in scenarios such as ICL prediction, where the relevant physical signals may lie across extended, diffuse regions and their boundaries.

To address previous limitations and preserve the ability to model long-range dependencies more efficiently, we turn our attention to an emerging class of models that offer an alternative to attention-based architectures. Over recent years, Deep State Space Models, such as Mamba (33), have appeared as a solid alternative to transformer based architectures, achieving similar or even better performance on sequence to sequence tasks while scaling competitively and being more efficient to train ($O(n \log n)$ with respect to the input size). For these reasons, and to leverage long range dependencies in velocity dispersion maps, we adapt State Space Models for ICL prediction.

State Space Models define a time invariant map $x(t) \in \mathbb{R} \mapsto y(t) \in \mathbb{R}$ through a hidden state $h(t) \in \mathbb{R}^N$. In the continuous case, this can be formulated as an Ordinary Differential Equation (ODE):

$$h'(t) = Ah(t) + Bx(t),$$
$$y(t) = Ch(t) + Dx(t),$$

for matrix $A \in \mathbb{R}^{N \times N}$ containing the evolution parameters, $B \in \mathbb{R}^N$, $C \in \mathbb{R}^N$ projection matrices and $D \in \mathbb{R}^1$ which functions as a residual connection. Here $y(t)$ corresponds to the output and $h'(t)$ to the evolution of the hidden state.

Drawing inspiration from the above formulation, structured state space sequence models (SSMs from now on) discretize the space for sequence modeling, in a way akin to Recurrent Neural Networks:

$$h_t = A_t h_{t-1} + B_t x_t, \tag{1}$$
$$y_t = C_t h_t + D_t x_t. \tag{2}$$

In this case, the matrix A is structured, meaning it is forced to be of diagonal form to enable efficient computation. Further, parameter matrices are made to be time-dependent, allowing the model to focus on different

parts of the input in different time-steps. These type of models are called *selective SSMs*, first introduced as the *Mamba* architecture by Gu and Dao (2023).

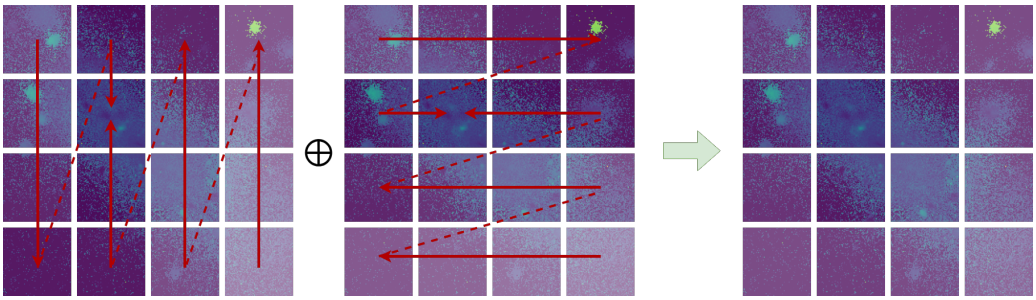


Figure 10: Adapting State Space Models to work with 2D sequences: the input is patched and traversed in four directions to produce sequences which are then fed to a SSM block. Results are aggregated at the end. Opacity is proportional to the amount of transformations applied to the hidden state after first seeing that patch.

This is the formulation we follow in our work, adapted to handle 2D sequences (namely, velocity dispersion maps). To do that, we first break the image into patches. Then we order these patches into sequences along four distinct traversal paths (left-to-right, top-to-bottom, right-to-left, bottom-to-top), and feed each sequence to a *selective SSM* block (see Figure 10). The resulting sequences are reshaped and merged to form the output map. In this manner, each pixel/patch integrates information from other pixels across different directions, allowing the simultaneous intervention of long range and local range dependencies in the computation of the output.

An overview of the functioning of 2D *selective SSM* blocks can be seen in Figure 11.

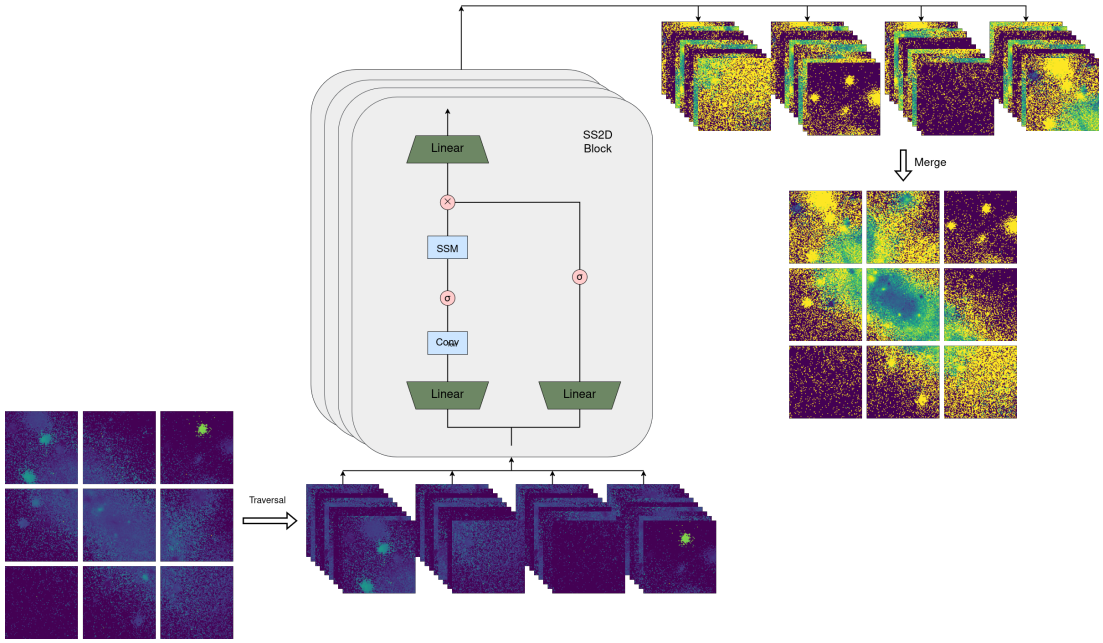


Figure 11: Overview of the functioning of a 2D *selective SSM* block (SS2D). In addition to the sequence processing units (SSM), which parametrize equations (1) and (2), the block contains linear + convolutional projections, as well as residual connections, which function on sequences of patches coming from each transversal direction. In our architecture, SS2D blocks are concatenated one after the other in an encoder-decoder structure. After every two blocks, feature dimensionality is reduced via patch merging in the encoder, and augmented via patch expanding in the decoder to allow for hierarchical representation learning of VD features.

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

2.2.4 Evaluation

During training, we observe that a reconstruction loss such as Pixel-wise Cross Entropy is most effective (see table 2). However, in order to fully capture cross-channel information, such as interloper masks, as well as evaluating local information reconstruction, we include two additional metrics for evaluation: Root Mean Squared Error (RMSE) (computed using a sliding window approach), and Relative Average Spectral Error (RASE), which is defined as follows:

$$RASE = \frac{100}{\mu}\sqrt{\frac{1}{N}\sum_{i=0}^{N} RMSE(B_i)^2}$$

where μ is the per pixel average, N the number of channels, $RMSE$ is the Robust Mean Squared Error computed using sliding window, and B_i corresponds to channel i.

RASE provides a relative and normalized error across feature channels, making it particularly useful in scenarios like ours where features vary in scale and intensity. By normalizing with respect to the mean signal, RASE penalizes models that fail to reconstruct small but physically important features while avoiding overemphasis on high-intensity regions. This makes it well-suited for evaluating models on astrophysical maps where subtle signals—such as low surface brightness structures or boundary transitions between BCG and ICL—are critical to the scientific interpretation.

In addition to this, we also report RMSE, as well as ICL fraction difference (or Mean Absolute Error) following the literature (28).

Besides these metrics of ICL predictive performance, we study the computational cost of each model, as well as scaling capabilities. Computational cost is computed as TFLOPs per forward pass, and train TFLOPs is computed taking into account the common estimation:

$$\text{FLOPs}_{\text{backward}} = 2.5 \times \text{FLOPs}_{\text{forward}}$$

Using this, we compute scaling curves of different model architectures, comparing training TFLOPs on an increasing number of halos with predictive performance on maps from a different simulation, showing how handling cross-simulation data leads to better generalization capabilities.

We conduct the following experiments:

1. **State-of-the-art reproduction**: Train and evaluation on the Dianoga dataset, consisting on 6241 generated maps from 8 halos, reproducing state-of-the-art work (Marini et al. (2024)) and adding new models for comparison.

2. **Generalization capabilities across simulations**: To evaluate whether results generalize across simulations, we evaluate models trained on maps from TNG300 and Eagle against a test set of unseen Eagle+TNG300 halos, as well as Dianoga halos.

3. **Scalability**: To evaluate whether scaling the amount of halos leads to better results, we train models on a huge combined dataset, containing 667605 maps generated from 3072 halos from the IllustrisTNG300 and Eagle simulations, and compare train-TFLOPs with performance on unseen simulations.

2.2.5 Results

State-of-the-art reproduction: We compare previous state-of-the art models (Baseline UNet and Baseline AttUNet) with improvements of these models (changing scheduler and objective), as well as new architectures: TransUNet and Mamba. We observe the best results are achieved by the improved UNet, while Mamba is competitive with a much lower computational cost. See Table 2.

Generalization capabilities across simulations: Using the huge dataset gathered in the previous stage, which contains over 3000 halos from 4 simulations, we train models on over 2600 halos (570k maps) from the TNG300 and Eagle simulations and evaluate on a test set of about 300 halos from the TNG300 and Eagle simulations. We observe models converge to similar performances (TransUNet being the exception). See Table 3.

We then evaluate those models on the completely unseen Dianoga dataset. See Table 4. Results are worse, as expected, but we note they match previous state-of-the-art models that were actually trained on that dataset. Further, we add RASE results after 1 single fine-tuning epoch on the Dianoga training set, showing models can be adapted to unseen simulations with very little compute.

Model	RASE ↓	RMSE ↓	ICL Frac ↓	Params (M)	TFLOPs
Baseline (UNet)	547.7	0.052	0.0436	31.0	0.023
Baseline (AttUNet)	544.7	0.052	0.0439	31.0	0.023
UNet	**385.3**	**0.036**	**0.0286**	31.0	0.023
Attention-UNet	418.3	0.040	0.0322	34.9	0.033
TransUNet	595.4	0.057	0.0490	13.3	0.004
Mamba (ps 1)	398.4	0.038	0.0295	11.5	0.033
Mamba (ps 4)	398.6	0.038	0.0295	11.5	**0.002**

Table 2: Model evaluation on the Dianoga dataset (lower is better). Metrics are Relative Average Spectral Error (RASE), Robust Mean Squared Error (RMSE). ICL Frac corresponds to ICL fraction absolute difference between ground truths and predictions, or Mean Absolute Error. TFLOPs are computed per forward pass on a single sample.

Model	RASE ↓	RMSE ↓	ICL Frac ↓	Params (M)	TFLOPs
UNet	76.32	0.006	0.0040	31.0	0.023
Attention-UNet	**75.91**	0.006	0.0040	34.9	0.033
TransUNet	146.8	0.012	0.0084	13.3	0.004
Mamba (ps 4)	79.77	0.006	0.0042	11.5	**0.002**

Table 3: Model evaluation on the unseen Dianoga dataset of models trained on a huge dataset containing only halos from TNG300 and Eagle. RASE-FT shows the performance obtained after a single fine-tuning epoch on 6 halos from the Dianoga dataset.

Model	RASE ↓	RASE (1 epch) ↓	RMSE ↓	ICL Frac ↓	Params (M)	TFLOPs
UNet	585.01	-	0.055	0.0441	31.0	0.023
Attention-UNet	**565.1**	-	**0.053**	**0.0427**	34.9	0.033
TransUNet	626.2	-	0.059	0.0510	13.3	0.004
Mamba (ps 4)	589.1	450.8	0.056	0.0449	11.5	**0.002**

Table 4: Model evaluation on the unseen Dianoga dataset of models trained on a huge dataset containing only halos from TNG300 and Eagle. RASE (1 epch) shows the performance obtained after a single fine-tuning epoch on 6 halos from the Dianoga dataset.

Scalability: We compute scaling laws for all compared models. We observe Mamba architectures scale better when compared with UNet based architectures, hinting at the possibility of further increasing the amount of training steps with new simulation data to increase performance. See Figure 12.

2.2.6 Limitations

Despite the strengths of our current pipeline, there are a few limitations worth noting. Working with 2D projections means we lose part of the 3D spatial and velocity information, which could otherwise provide richer context for distinguishing between ICL and BCG regions. Our labeling approach also depends on a Random Forest model trained on a relatively small sample of halos from the DIANOGA simulation, which may not generalize perfectly across different simulation suites with varying resolutions and physical models.

2.2.7 Work plan for release #2

In the upcoming release, we aim to extend the dataset generation pipeline to cover all targeted simulations, enabling a more comprehensive training set for model development. A key goal is to train a foundational deep learning model that generalizes across simulation domains while capturing shared physical structures. In order to deal with the loss of 3D information resulting from working with projections, we are planning on introducing both 3D and 2D encoders trained with a contrastive loss, which would allow making inference directly from 2D observational data while leveraging the 3D nature of simulation data. We also plan to experiment with the inference of additional properties beyond the ICL, including the use of kinematic information to predict other quantities such as the dark matter distribution in both 2D and 3D. To enrich the learning signal and improve

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

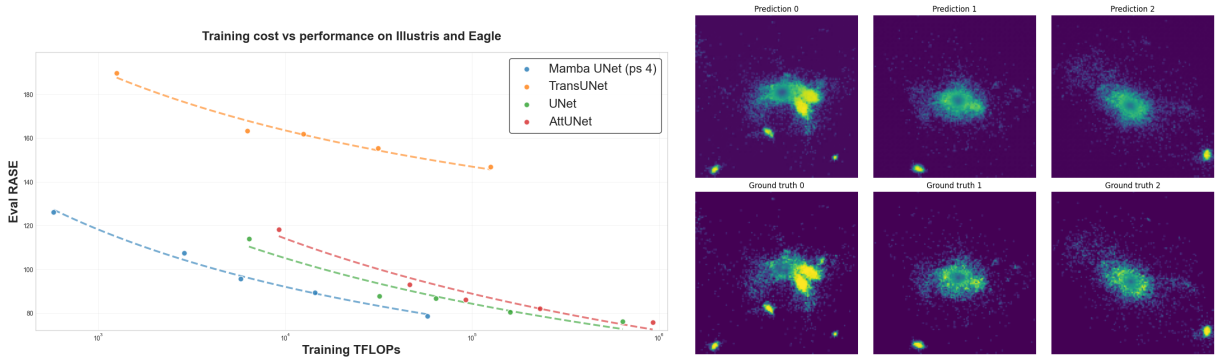


Figure 12: (Left) Scaling laws computed for all architectures on 3091 halos from the Illustris and Eagles dataset. Mamba scales comparatively better than other models. (Right) ICL predictions using a Mamba architecture.

model robustness, we will incorporate multi-channel inputs, such as interloper masks and uncertainty scores derived from the Random Forest classifier. These additions will allow the model to account for observational contaminants and prediction confidence. Lastly, model evaluation will be expanded using a custom RASE metric tailored to the spatial and statistical characteristics of the target maps, providing a more physically meaningful assessment of model performance.

In addition, we will explore the use of IFU maps and other observables as inputs to predict other physical properties of galaxies and clusters that are inaccessible through observations, including for instance, the 2D/3D dark matter distribution. By leveraging these kinematic features, the models can be extended beyond ICL prediction and applied to a broader range of astrophysical problems, including the study of structure formation and galaxy evolution.

In parallel, we are preparing a scientific publication in collaboration with Ilaria Marini and members of the hydrodynamical simulation community, and will present our results at the upcoming EAS (European Astronomical Society) Annual Meeting.

2.3 Predicting the effect of radiation on gas cooling rates on-the-fly in hydrodynamical simulations

Lead WP3 partner: BSC

2.3.1 Introduction

Cooling rates play a fundamental role in a wide range of astrophysical scenarios, including galaxy formation and the thermal evolution of the universe following the Big Bang. However, accurate computation of these rates presents significant challenges, as the resulting values are highly sensitive to the underlying physical assumptions, such as the adoption of collisional ionization equilibrium (CIE), which can lead to substantial variability in outcomes.

Traditionally, in hydrodynamical codes such as OpenGadget3 (34), cooling rates cooling rates of the plasma are represented through a total cooling rate using Equation 3

$$\Lambda = \Lambda_{H,He} + \sum_{i>He} \Lambda_{i,\odot} \left(\frac{n_e}{n_{e,\odot}}\right) \left(\frac{n_i}{n_{i,\odot}}\right) = \Lambda_{H,He} + \sum_{i>He} \Lambda_{i,\odot} \frac{n_e/n_H}{(n_e/_H)_\odot} 10^{[i/H]} \tag{3}$$

where the elemental cooling rates are obtained through interpolation of pre-computed tabulated data from other codes that assume CIE such as CLOUDY. Although this method is widely used, it incurs high computational costs and limits flexibility in model adaptation. To address these limitations in hydrodynamical codes and streamline the input requirements, the total cooling rate of the plasma will instead be calculated using Equation 3, enabling a more efficient and generalizable approach. To address the computational complexity associated with the interpolation of the cooling rates from the tables, a surrogate model is proposed to approximate the cooling contributions of the elements present in the simulation and to provide a general estimate of the total cooling rate. The model operates through two distinct approaches, depending on the type of element being considered.

For hydrogen (H) and helium (He), the cooling rate is obtained directly from the tabulated data from CLOUDY and is a function of four key variables, which are described in detail in the following section. A dedicated dataset, generated from these tables, will be used to train the surrogate model for this component. This part of the model has a critical impact on the overall accuracy and acceptability of the global surrogate solution.

For metals, individual contributions to the total cooling rates are calculated using solar reference values ($\Lambda_{i,\odot}$). These values are also tabulated and depend on three variables, similar to those used for hydrogen and helium, but exclude the dependence on the helium abundance (n_{He}/n_H). The contribution of each element is independently evaluated and incorporated into the total cooling estimate, ensuring consistency and scalability in varying chemical compositions. An additional component required to model the cooling contribution of metals is the ratio of free electrons to hydrogen atoms (n_e/n_H), which is computed for both general and solar abundances. This value is retrieved from precomputed tables and is dependent on the same three variables used in the solar cooling rate calculations. The contribution of the Solar component to the total cooling rate is shown in Figure 13.

The complete set of elements included in the surrogate model is drawn from the cooling rates computed by CLOUDY and is referred to as the x_i group. This group comprises the following elements: carbon (C), nitrogen (N), oxygen (O), neon (Ne), magnesium (Mg), silicon (Si), sulfur (S), calcium (Ca), iron (Fe), sodium (Na), aluminum (Al), argon (Ar) and nickel (Ni).

Although tabulated cooling data for a wider range of elements could be obtained, only a subset has been selected for inclusion in the current model. Elements with lower abundances in the plasma have smaller contributions to the overall cooling rate, and their inclusion would unnecessarily increase the model complexity at the first stage. The addition of elements with lower abundances and other parameters is left for a second stage revision of the model.

Additionally, the model incorporates a dependency on metallicity (Z). For the first test implementation of the model, a fixed value of $Z = 1$ has been assumed to simplify the training and evaluation process. However, this assumption will be relaxed in future iterations to allow the model to accommodate varying metallicities and potentially include a broader set of elements from the total metals group.

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

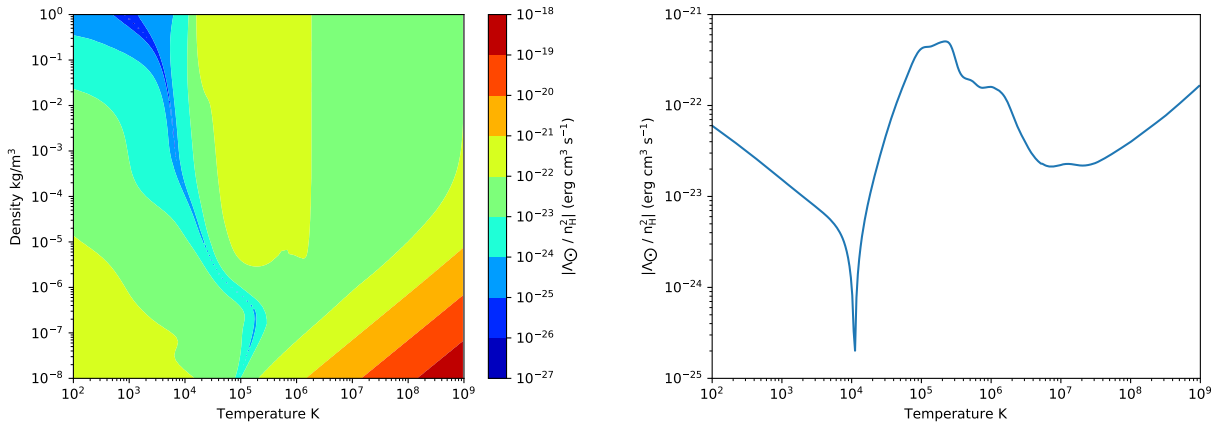


Figure 13: On both plots: normalized, absolute, net cooling rates for the Solar component ($\Lambda_{i,\odot}$) for redshift zero. The left plot shows the cooling rates as a function of density and temperature while on the right shows a horizontal cut of the left plot, i.e., the cooling rates as a function of the temperature at a fixed density.

2.3.2 Tool development

Our approach uses the pyLOM framework (35) to construct a surrogate model to approximate the gas cooling rates (Λ) from data computed by the astrophysical spectral synthesis code CLOUDY thus mimicking the work of Wiersma et al. (36). All the cooling rates in the tables were computed by running large grids of photoionization models, which contain the values that are considered relevant for the temperature of gas in cosmological simulations (from 10^4 to 10^8). The assumptions in the computed gas cooling rate tables are the following:

1. exposed to cosmic microwave background radiation (CMB),
2. exposed to the Haardt & Madau 2001 (37) model for the UV/X-ray background radiation,
3. dust-free,
4. optically thin,
5. in ionization equilibrium.

All cooling rates are tabulated as a function of the total hydrogen number density $\log nH$ (see Equation 1 in Wiersma et al.), the temperature ($\log T$), in Kelvin, and the redshift z which encodes the time-dependence of the metagalactic ionizing background. The surrogate model is designed to take these key physical inputs along with helium abundance (n_{He}/n_H) and metallicity (Z) among possible others. To learn the complex, nonlinear mapping between the input gas properties and the cooling rates, a MLP is employed based on the implementation in PyTorch, offering a flexible and efficient neural network architecture suitable for regression tasks of this kind.

The next step is to replace the traditional multidimensional interpolation tables currently used in cosmological simulation codes with this PyTorch-based surrogate model. Using the torch C/C++ and Fortran interface, PyTorch models can be directly loaded and evaluated inside the compiled code. Thus, our model can be directly connected to them, allowing seamless integration. At this point, tests will be conducted to verify that the new model enhances and accelerates evaluations.

The end goal is to obtain on-the-fly training and evaluation of the model: if the code queries cooling rates for input combinations outside the surrogate trained domain, those points can be flagged, and new data can be generated from CLOUDY (38) to retrain and extend the model. This iterative process will improve accuracy over time, ensuring that the surrogate model remains valid under a wide range of physical conditions. A flowchart diagram of the ML workflow is presented in Figure 14.

Data Loading: As outlined in the tool development section, the primary data calculated using CLOUDY include the cooling rates as functions of hydrogen number density and temperature. These parameters form

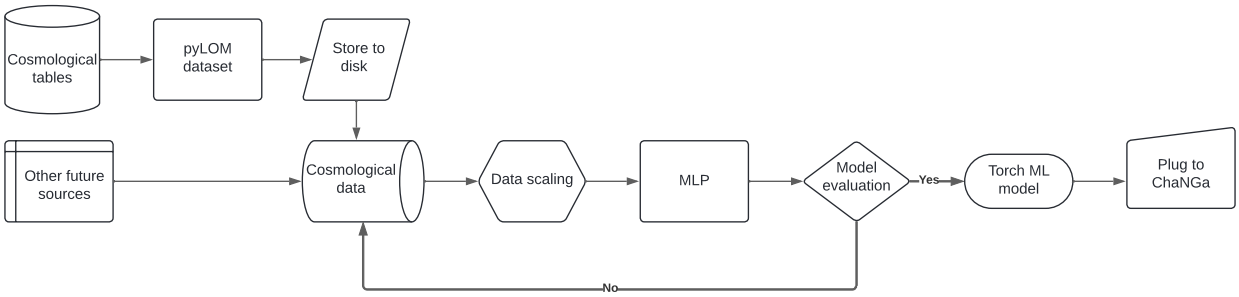


Figure 14: Flowchart diagram of the ML model workflow.

the foundation of the input data for the surrogate model. The dataset comprises 49 structured tables, each corresponding to a different redshift (z) value, ranging from 0 to 8.989. Despite varying redshifts, all tables share a consistent format, which facilitates automated data parsing and integration into the model pipeline.

Each table is organized into the following main sections: a header group, a metal-free group (for hydrogen and helium), individual values for each element in the x_i group, a solar abundance group, and a total metals group.

- From the header group, the redshift value corresponding to the current table is recovered, along with the total number of data points for each element bin and the species names used to identify the elements in the x_i group.
- From the metal-free group, the net cooling rates and associated variable bins are retrieved. These currently serve as direct input for the model.
- For each element in the x_i group, the net cooling rates and the corresponding variable bins are extracted, enabling the construction of individual surrogate models per element.
- From the solar group, the same variable bins used for the x_i elements are retrieved. In addition, the electron-to-hydrogen number density ratio $(n_e/n_H)_\odot$ is obtained, which depends on temperature and hydrogen number density.
- From the total metals group, the same variable bins as those used for individual x_i elements are extracted.

Data Scaling: The input data for neural networks must be scaled between 0 and 1 to undergo training. Any data above 1 or below 0 will be understood as its limiting value by the network. Thus, the data is scaled using a *minmax* approach as

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}}(\text{feature\_range}_{max} - \text{feature\_range}_{min}) + \text{feature\_range}_{min}, \tag{4}$$

where X_{scaled} is the scaled version of the input data in the range between $\text{feature\_range}_{min}$ and $\text{feature\_range}_{max}$, X is the input data with X_{max} and X_{min} its maximum and minimum respectively. In this case, $\text{feature\_range}_{max} = 1$ and $\text{feature\_range}_{min} = 0$ to scale the data between 0 and 1. However, this scaling is not enough to successfully generate training on the data as it is very compressed. To compensate for this effect, a logarithmic scaling is performed on X before the *minmax* scaling as

$$X = \log\left(Y + 1.01\,|Y_{min}|\right), \tag{5}$$

where Y is the original data from the tables. The term $1.01\,|Y_{min}|$ is included to avoid the logarithm of 0.

2.3.3 Results

The first training tests for the MLP were performed for the metal-free and solar groups at redshift zero for a fixed nH. Training was carried out for 1500 epochs to ensure a correct representation of the function, and the data was split 80%/20% for training and testing. The results for this case are presented in Figure 15, showing

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

that the cooling rate is correctly predicted by the MLP with $R^2 = 0.96$ and $RMSE = 0.0187$ for the metal-free group, and $R^2 = 0.99$ and $RMSE = 0.0146$ for the Solar group. Looking at the training loss in Figure 16, it can be seen that the model trains correctly and that the MLP correctly captures the data. Discrepancies arise due to the lack of data in the region between 0.4 and 0.8 of the scaled temperature, as seen in Figure 17. This is to be expected in this first test and will be reduced once more data is pooled in. This first test of concept proves that the MLP can correctly approximate the cooling rate.

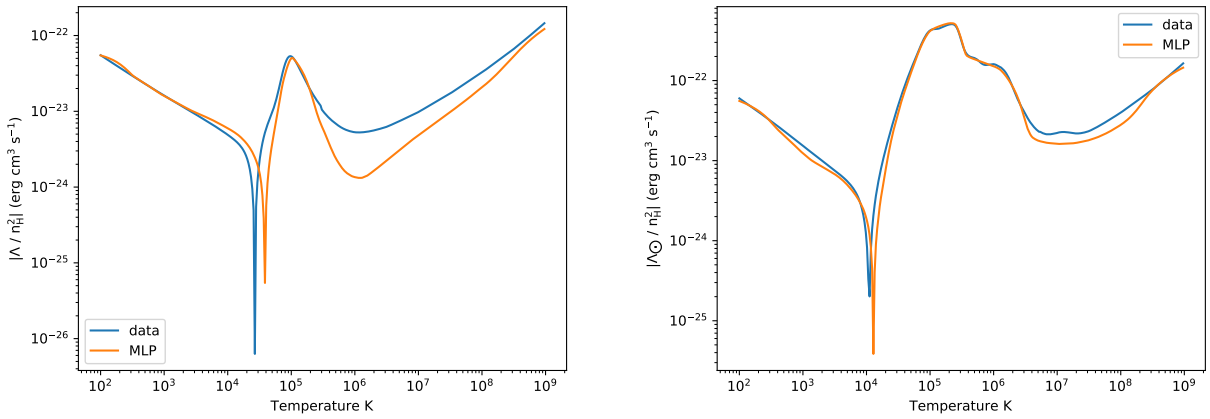


Figure 15: MLP results on the metal-free group (left) and Solar group (right).

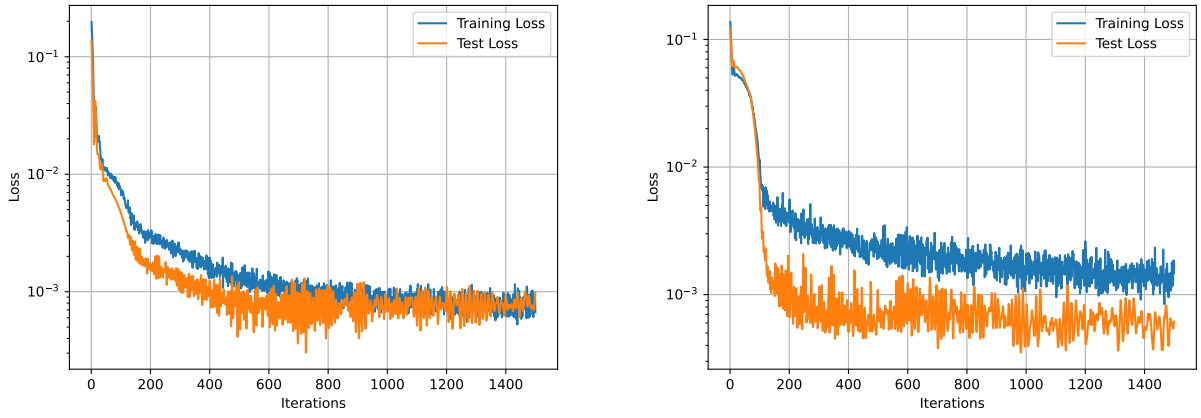


Figure 16: Training loss for the metal-free group (left) and Solar group (right).

The next step was to add the dependency of the total hydrogen number density at redshift zero. The MLP parameters were kept as the previous test with the same data split of 80%/20% for training and testing. The results for this case are presented in Figure 18 for the metal-free group (top) with $R^2 = 0.98$ and Solar group (bottom) with $R^2 = 0.97$, respectively. Again, it is proven that the model trains correctly and that the MLP correctly captures the data. Moreover, the addition of extra data in the training process allows for a better representation of the surrogate ML model and a faster training of the model.

2.3.4 Work plan for release #2

As part of release #2, the MLP will be consolidated as the surrogate machine learning model to predict the total cooling rate Λ based on existing datasets and standard input parameters. We will focus specifically on the metal-free group at redshift zero and its integration into the OpenGadget3 (34) code. This will enable simulations of an isolated galaxy with hydrogen and helium only. It will also allow us to close the loop and perform a

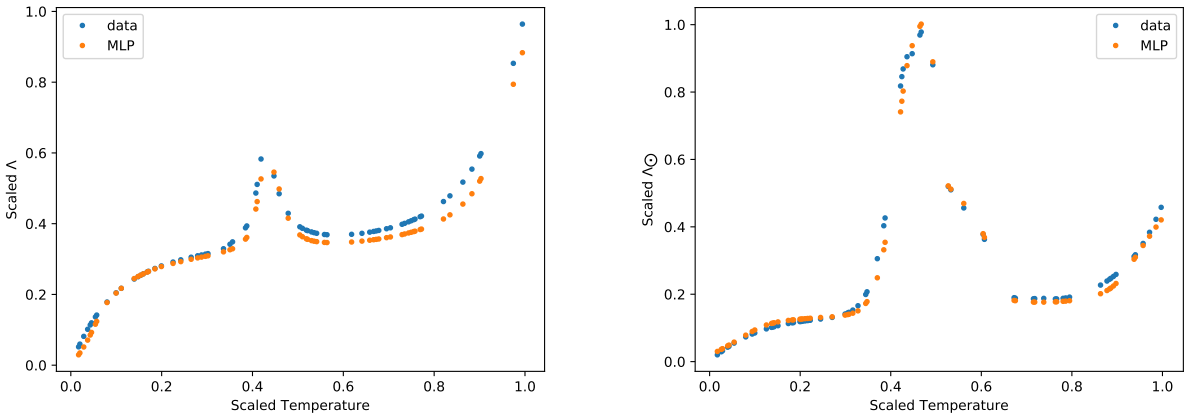


Figure 17: Scaled data for the metal-free group (left) and Solar group (right).

comparison between the classical interpolation and the MLP, both in terms of accuracy and computational time.

Following this first test, the full model will be integrated into OpenGadget3, where it is intended to replace the current multidimensional interpolation routine. This integration will be carried out using the Torch C/C++ interface, though potential performance bottlenecks may arise due to the data handling requirements of Torch. As part of this process, an evaluation will be conducted to determine whether batch processing of input data or point-wise model evaluation yields better performance.

In parallel, efforts will be made to update the model by exploring the inclusion of additional input parameters available within OpenGadget3. The impact of these parameters on model performance will be assessed, and mechanisms will be investigated through which OpenGadget3 can generate new training points to enhance model learning. By the time of the next release, initial development work on model integration and evaluation strategies is expected to be underway.

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

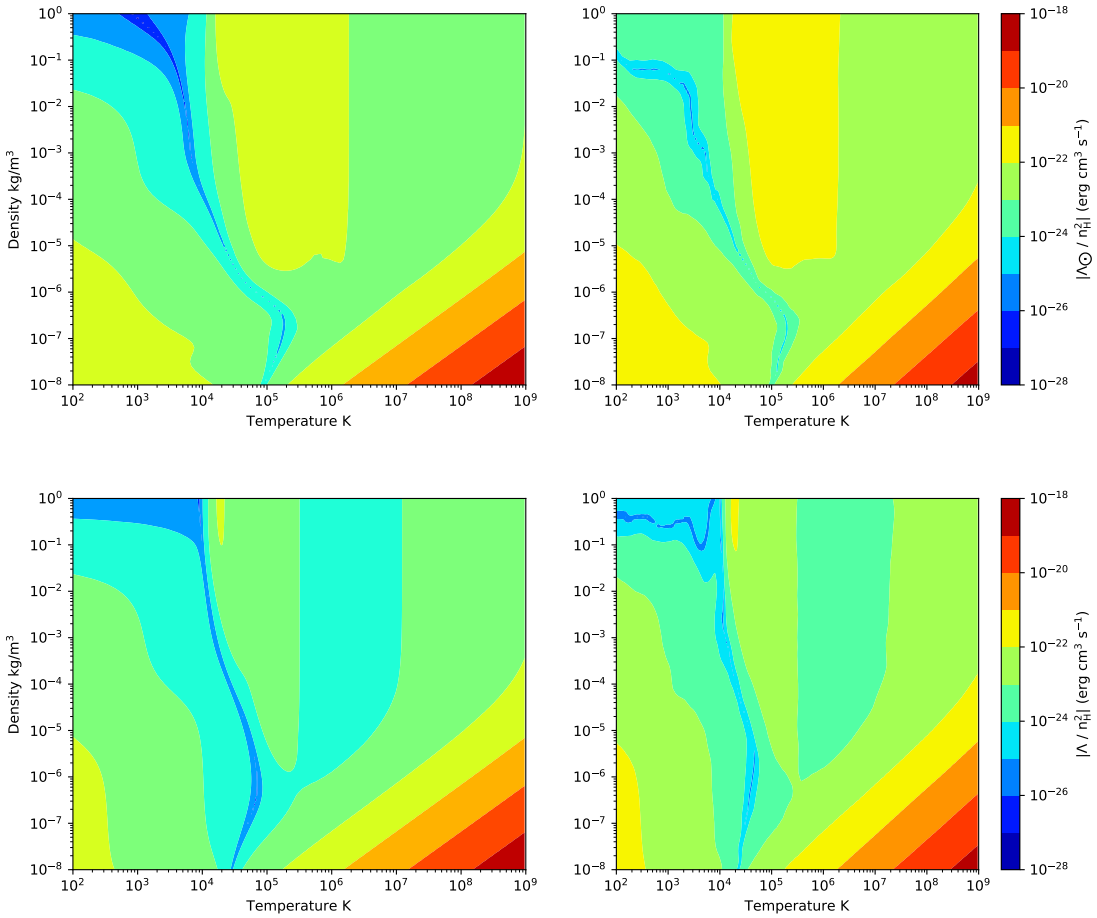


Figure 18: MLP results on the metal-free group (bottom) and Solar group (top). Left: original data. Right: results of the MLP evaluation.

3 Conclusions

The initial release of the SPACE CoE ML framework marks substantial progress toward fulfilling the objectives outlined in the SPACE project, which aims to integrate state-of-the-art data analysis techniques and machine learning methodologies into astrophysical and cosmological research to enhance scientific discovery through exascale numerical simulations. Specifically, it targets three critical use cases identified in D3.1: Representation Learning for Explorative Knowledge Discovery, Physics-informed Reconstruction from Synthetic Observations, and Predicting the Effects of Radiation on Gas Cooling Rates in hydrodynamical simulations. These objectives are strategically chosen to address complex challenges posed by the scale and complexity of cosmological simulation data, aiming to facilitate improved interpretability, computational efficiency, and scientific accuracy.

The Representation Learning component has effectively addressed the challenge of exploring and interpreting massive and high-dimensional simulation datasets. The Spherinator and HiPSter modules, utilizing hyperspherical variational autoencoders, significantly streamline the process of condensing intricate simulation outputs into compact, meaningful representations. This advancement aligns with the project's goals of improving exploratory capabilities and making simulation data more accessible and interpretable, thereby enhancing the potential for scientific discovery.

Physics-informed reconstruction has made meaningful progress through the use of synthetic observational data, especially in studying the intracluster light (ICL) in galaxy clusters. By leveraging advanced machine learning models—such as convolutional networks and transformer-based architectures — the framework can recover subtle astrophysical features from observational proxies. This approach offers a promising step toward narrowing the gap between simulations and real observations, helping us better understand the structure and

evolution of galaxy clusters.

Lastly, significant improvements were achieved in predicting radiation effects on gas cooling rates through a surrogate model built on Multi-Layer Perceptrons (MLPs). This method has shown considerable advantages over traditional interpolation techniques, offering increased computational efficiency and improved predictive accuracy. Moving forward, the project intends to further refine these ML tools, broaden their dataset scope, and integrate advanced methods to ensure scalability and accuracy across diverse astrophysical scenarios.

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

References

[1] I. Marini, S. Borgani, A. Saro, G. Murante, G. L. Granato, C. Ragone-Figueroa, and G. Taffoni, "Machine learning to identify ICL and BCG in simulated galaxy clusters," *Monthly Notices of the Royal Astronomical Society*, vol. 514, no. 2, pp. 3082–3096, Jun. 2022. [Online]. Available: https://academic.oup.com/mnras/article/514/2/3082/6604895

[2] P. Fernique, M. G. Allen, T. Boch, A. Oberto, F. X. Pineau, D. Durand, C. Bot, L. Cambrésy, S. Derriere, F. Genova, and F. Bonnarel, "Hierarchical progressive surveys. Multi-resolution HEALPix data structures for astronomical images, catalogues, and 3-dimensional data cubes," , vol. 578, p. A114, Jun. 2015.

[3] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne *et al.*, "The fair guiding principles for scientific data management and stewardship," *Scientific data*, vol. 3, 2016.

[4] N. D. Cao and W. Aziz, "The power spherical distribution," *Proceedings of the 37th International Conference on Machine Learning, INNF+*, 6 2020. [Online]. Available: http://arxiv.org/abs/2006.04437

[5] T. R. Davidson, L. Falorsi, N. D. Cao, T. Kipf, and J. M. Tomczak, "Hyperspherical variational auto-encoders," *34th Conference on Uncertainty in Artificial Intelligence (UAI-18)*, 4 2018. [Online]. Available: http://arxiv.org/abs/1804.00891

[6] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *Proceedings of the 2nd International Conference on Learning Representations (ICLR)*, 2014. [Online]. Available: https://arxiv.org/abs/1312.6114

[7] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908*, 2016. [Online]. Available: https://arxiv.org/abs/1606.05908

[8] A. Asperti and M. Trentin, "Balancing reconstruction error and kullback-leibler divergence in variational autoencoders," *IEEE Access*, vol. 8, pp. 199 440–199 448, 2 2020. [Online]. Available: http://arxiv.org/abs/2002.07514

[9] D. Nelson, A. Pillepich, S. Genel, M. Vogelsberger, V. Springel, P. Torrey, V. Rodriguez-Gomez, D. Sijacki, G. Snyder, B. Griffen, F. Marinacci, L. Blecha, L. Sales, D. Xu, and L. Hernquist, "The illustris simulation: Public data release," *Astronomy and Computing*, vol. 13, p. 12–37, Nov. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.ascom.2015.09.003

[10] D. Nelson, V. Springel, A. Pillepich, V. Rodriguez-Gomez, P. Torrey, S. Genel, M. Vogelsberger, R. Pakmor, F. Marinacci, R. Weinberger, L. Kelley, M. Lovell, B. Diemer, and L. Hernquist, "The illustristng simulations: Public data release," 2021. [Online]. Available: https://arxiv.org/abs/1812.05609

[11] V. Rodriguez-Gomez, G. F. Snyder, J. M. Lotz, D. Nelson, A. Pillepich, V. Springel, S. Genel, R. Weinberger, S. Tacchella, R. Pakmor, P. Torrey, F. Marinacci, M. Vogelsberger, L. Hernquist, and D. A. Thilker, "The optical morphologies of galaxies in the IllustrisTNG simulation: a comparison to Pan-STARRS observations," , vol. 483, no. 3, pp. 4140–4159, Mar. 2019.

[12] B. Doser, K. L. Polsterer, A. Fehlner, and S. Trujillo-Gomez, "Machine learning workflow for morphological classification of galaxies," 2025. [Online]. Available: https://arxiv.org/abs/2505.04676

[13] K. L. Polsterer, F. Gieseke, and C. Igel, "Automatic Galaxy Classification via Machine Learning Techniques: Parallelized Rotation/Flipping INvariant Kohonen Maps (PINK)," in *Astronomical Data Analysis Software an Systems XXIV (ADASS XXIV)*, ser. Astronomical Society of the Pacific Conference Series, A. R. Taylor and E. Rosolowsky, Eds., vol. 495, Sep. 2015, p. 81.

[14] M. Ho, D. J. Bartlett, N. Chartier, C. Cuesta-Lazaro, S. Ding, A. Lapel, P. Lemos, C. C. Lovell, T. L. Makinen, C. Modi, V. Pandya, S. Pandey, L. A. Perez, B. Wandelt, and G. L. Bryan, "Ltu-ILI: An All-in-One Framework for Implicit Inference in Astrophysics and Cosmology," *The Open Journal of Astrophysics*, vol. 7, jul 3 2024.

[15] P. Monaco, G. Murante, S. Borgani, and F. Fontanot, "Diffuse Stellar Component in Galaxy Clusters and the Evolution of the Most Massive Galaxies at [FORMULA][F]z1[/F][/FORMULA]," *The Astrophysical Journal*, vol. 652, no. 2, pp. L89–L92, Dec. 2006. [Online]. Available: https://iopscience.iop.org/article/10.1086/510236

[16] M. Montes, "The intracluster light and its role in galaxy evolution in clusters," Dec. 2019, arXiv:1912.01616 [astro-ph]. [Online]. Available: http://arxiv.org/abs/1912.01616

[17] F. Marinacci, M. Vogelsberger, R. Pakmor, P. Torrey, V. Springel, L. Hernquist, D. Nelson, R. Weinberger, A. Pillepich, J. Naiman, and S. Genel, "First results from the IllustrisTNG simulations: radio haloes and magnetic fields," *Monthly Notices of the Royal Astronomical Society*, Aug. 2018, arXiv:1707.03396 [astro-ph]. [Online]. Available: http://arxiv.org/abs/1707.03396

[18] V. Springel, R. Pakmor, A. Pillepich, R. Weinberger, D. Nelson, L. Hernquist, M. Vogelsberger, S. Genel, P. Torrey, F. Marinacci, and J. Naiman, "First results from the IllustrisTNG simulations: matter and galaxy clustering," *Monthly Notices of the Royal Astronomical Society*, vol. 475, no. 1, pp. 676–698, Mar. 2018, arXiv:1707.03397 [astro-ph]. [Online]. Available: http://arxiv.org/abs/1707.03397

[19] D. Nelson, A. Pillepich, V. Springel, R. Weinberger, L. Hernquist, R. Pakmor, S. Genel, P. Torrey, M. Vogelsberger, G. Kauffmann, F. Marinacci, and J. Naiman, "First results from the IllustrisTNG simulations: the galaxy color bimodality," *Monthly Notices of the Royal Astronomical Society*, vol. 475, no. 1, pp. 624–647, Mar. 2018, arXiv:1707.03395 [astro-ph]. [Online]. Available: http://arxiv.org/abs/1707.03395

[20] A. Pillepich, D. Nelson, L. Hernquist, V. Springel, R. Pakmor, P. Torrey, R. Weinberger, S. Genel, J. Naiman, F. Marinacci, and M. Vogelsberger, "First results from the IllustrisTNG simulations: the stellar mass content of groups and clusters of galaxies," *Monthly Notices of the Royal Astronomical Society*, vol. 475, no. 1, pp. 648–675, Mar. 2018, arXiv:1707.03406 [astro-ph]. [Online]. Available: http://arxiv.org/abs/1707.03406

[21] J. P. Naiman, A. Pillepich, V. Springel, E. Ramirez-Ruiz, P. Torrey, M. Vogelsberger, R. Pakmor, D. Nelson, F. Marinacci, L. Hernquist, R. Weinberger, and S. Genel, "First results from the IllustrisTNG simulations: A tale of two elements – chemical evolution of magnesium and europium," *Monthly Notices of the Royal Astronomical Society*, vol. 477, no. 1, pp. 1206–1224, Jun. 2018, arXiv:1707.03401 [astro-ph]. [Online]. Available: http://arxiv.org/abs/1707.03401

[22] L. Bassini, E. Rasia, S. Borgani, G. L. Granato, C. Ragone-Figueroa, V. Biffi, A. Ragagnin, K. Dolag, W. Lin, G. Murante, N. R. Napolitano, G. Taffoni, L. Tornatore, and Y. Wang, "The DIANOGA simulations of galaxy clusters: characterising star formation in protoclusters," *Astronomy & Astrophysics*, vol. 642, p. A37, Oct. 2020. [Online]. Available: https://www.aanda.org/10.1051/0004-6361/202038396

[23] K. Dolag, "The Magneticum Simulations, from Galaxies to Galaxy Clusters," vol. 29, Aug. 2015, p. 2250156, aDS Bibcode: 2015IAUGA..2250156D. [Online]. Available: https://ui.adsabs.harvard.edu/abs/2015IAUGA..2250156D

[24] J. Schaye, R. Kugel, M. Schaller, J. C. Helly, J. Braspenning, W. Elbers, I. G. McCarthy, M. P. v. Daalen, B. Vandenbroucke, C. S. Frenk, J. Kwan, J. Salcido, Y. M. Bahé, J. Borrow, E. Chaikin, O. Hahn, F. Huško, A. Jenkins, C. G. Lacey, and F. S. J. Nobels, "The FLAMINGO project: cosmological hydrodynamical simulations for large-scale structure and galaxy cluster surveys," *Monthly Notices of the Royal Astronomical Society*, vol. 526, no. 4, pp. 4978–5020, Oct. 2023, arXiv:2306.04024 [astro-ph]. [Online]. Available: http://arxiv.org/abs/2306.04024

[25] R. Pakmor, V. Springel, J. P. Coles, T. Guillet, C. Pfrommer, S. Bose, M. Barrera, A. M. Delgado, F. Ferlito, C. Frenk, B. Hadzhiyska, C. Hernández-Aguayo, L. Hernquist, R. Kannan, and S. D. M. White, "The MillenniumTNG Project: The hydrodynamical full physics simulation and a first look at its galaxy clusters," *Monthly Notices of the Royal Astronomical Society*, vol. 524, no. 2, pp. 2539–2555, Jul. 2023, arXiv:2210.10060 [astro-ph]. [Online]. Available: http://arxiv.org/abs/2210.10060

[26] S. McAlpine, J. C. Helly, M. Schaller, J. W. Trayford, Y. Qu, M. Furlong, R. G. Bower, R. A. Crain, J. Schaye, T. Theuns, C. D. Vecchia, C. S. Frenk, I. G. McCarthy, A. Jenkins, Y. Rosas-Guevara,

This document is Public (PU)
and was produced under SPACE project EU GA 101093441

S. D. M. White, M. Baes, P. Camps, and G. Lemson, "The EAGLE simulations of galaxy formation: public release of halo and galaxy catalogues," *Astronomy and Computing*, vol. 15, pp. 72–89, Apr. 2016, arXiv:1510.01320 [astro-ph]. [Online]. Available: http://arxiv.org/abs/1510.01320

[27] K. Dolag, G. Murante, and S. Borgani, "Dynamical difference between the cd galaxy and the diffuse, stellar component in simulated galaxy clusters," *Monthly Notices of the Royal Astronomical Society*, vol. 405, no. 3, pp. 1544–1559, 2010.

[28] I. Marini, A. Saro, S. Borgani, and M. Boi, "Inferring intrahalo light from stellar kinematics: A deep learning approach," *Astronomy & Astrophysics*, vol. 689, p. A181, Sep. 2024. [Online]. Available: https://www.aanda.org/10.1051/0004-6361/202449632

[29] I. Marini, S. Borgani, A. Saro, G. L. Granato, C. Ragone-Figueroa, B. Sartoris, K. Dolag, G. Murante, A. Ragagnin, and Y. Wang, "Velocity dispersion of the brightest cluster galaxies in cosmological simulations," *Monthly Notices of the Royal Astronomical Society*, vol. 507, no. 4, pp. 5780–5795, Sep. 2021, arXiv:2109.00223 [astro-ph]. [Online]. Available: http://arxiv.org/abs/2109.00223

[30] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," 2015, version Number: 1. [Online]. Available: https://arxiv.org/abs/1505.04597

[31] O. Oktay, J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, B. Glocker, and D. Rueckert, "Attention U-Net: Learning Where to Look for the Pancreas," 2018, version Number: 3. [Online]. Available: https://arxiv.org/abs/1804.03999

[32] J. Chen, Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, and Y. Zhou, "Transunet: Transformers make strong encoders for medical image segmentation," *arXiv preprint arXiv:2102.04306*, 2021.

[33] A. Gu and T. Dao, "Mamba: Linear-Time Sequence Modeling with Selective State Spaces," 2023, version Number: 2. [Online]. Available: https://arxiv.org/abs/2312.00752

[34] K. Dolag et al., "Manuscript in preparation," 2025.

[35] B. Eiximeno, A. Miró, B. Begiashvili, E. Valero, I. Rodriguez, and O. Lehmkhul, "PyLOM: A HPC open source reduced order model suite for fluid dynamics applications," *Computer Physics Communications*, vol. 308, p. 109459, 2025.

[36] R. P. C. Wiersma, J. Schaye, and B. D. Smith, "The effect of photoionization on the cooling rates of enriched, astrophysical plasmas," , vol. 393, no. 1, pp. 99–107, Feb. 2009.

[37] F. Haardt and P. Madau, "Modelling the uv/x-ray cosmic background with cuba," 2001. [Online]. Available: https://arxiv.org/abs/astro-ph/0106018

[38] G. Ferland, M. Chatzikos, F. Guzmán, M. Lykins, P. Van Hoof, R. Williams, N. Abel, N. Badnell, F. Keenan, R. Porter *et al.*, "The 2017 release of cloudy," *Revista mexicana de astronomía y astrofísica*, vol. 53, no. 2, 2017.