Characterizing structure formation through instance segmentation

Daniel López-Cano,[1,2★] Jens Stücker,[1] Marcos Pellejero Ibañez,[3] Raúl E. Angulo,[1,4]
Daniel Franco-Barranco[1,5]
[1] Donostia International Physics Center (DIPC), Paseo Manuel de Lardizabal, 4, 20018 Donostia-San Sebastián, Spain
[2] Departamento de Física Teórica, Módulo 15, Facultad de Ciencias, Universidad Autónoma de Madrid (UAM), 28049 Madrid, Spain
[3] Institute for Astronomy, University of Edinburgh, Royal Observatory, Blackford Hill, Edinburgh, EH9 3HJ , UK
[4] IKERBASQUE, Basque Foundation for Science, 48013, Bilbao, Spain.
[5] Department of Computer Science and Artificial Intelligence, University of the Basque Country (UPV/EHU), Donostia-San Sebastián, Spain

Accepted XXX. Received YYY; in original form ZZZ

ABSTRACT

Dark matter haloes form from small perturbations to the almost homogeneous density field of the early universe. Although it is known how large these initial perturbations must be to form haloes, it is rather poorly understood how to predict which particles will end up belonging to which halo. However, it is this process that determines the Lagrangian shape of protohaloes and is therefore essential to understand their mass, spin and formation history. Here, we present a machine learning framework to learn how the protohalo regions of different haloes emerge from the initial density field. This involves one neural network to distinguish semantically which particles become part of *any* halo and a second neural network that groups these particles by halo membership into different instances. This instance segmentation is done through the Weinberger method, in which the network maps particles into a pseudo-space representation where different instances can be distinguished easily through a simple clustering algorithm. Our model reliably predicts the masses and Lagrangian shapes of haloes object-by-object, as well as summary statistics like the halo-mass function. We find that our model extracts information close to optimal by comparing it to the degree of agreement between two N-body simulations with slight differences in their initial conditions. We publish our model open-source and suggest that it can be used to inform analytical methods of structure formation by studying the effect of systematic manipulations of the initial conditions.

Key words: methods: numerical, statistical, data analysis – cosmology: dark matter

1 INTRODUCTION

Dark matter (DM) haloes are the primary structures in the universe within which galaxies form and evolve. Acting as gravitational anchors, they play a pivotal role in connecting theoretical cosmology with empirical observations from galaxy surveys. Given their significance in cosmology, a comprehensive understanding of DM haloes and their behavior is paramount. Currently, our most detailed insights into their formation and properties come from N-body simulations (see Frenk & White 2012, for a review). These computationally-intensive simulations model the interactions of vast numbers of particles, pinpointing the regions of the density field where gravitational collapse leads to the formation of DM haloes (e.g. Angulo & Hahn 2022). Therefore, understanding the formation and behavior of DM haloes is essential to bridge the gap between theoretical models and observational data.

However, providing quick and accurate predictions (based on the initial conditions of a simulation) remains a challenging task for physically-motivated models. An accurate model for halo formation must be able to capture the nonlinear growth of density fluctuations. Previous analytical or semi-analytical models for halo formation, such as the top-hat spherical collapse (Gunn & Gott 1972; Gunn 1977; Peebles 1980), the Press-Schechter / Excursion Set Theory (Press & Schechter 1974; Bond et al. 1991; Lacey & Cole 1993), or ellipsoidal collapse approaches (e.g. Sheth et al. 2001; Sheth & Tormen 2002), qualitatively reproduce the behaviour of the halo-mass function and the merging rate of haloes, however, they fail on predicting these quantities accurately (e.g. Jiang & van den Bosch 2014). Further, N-body simulations show the formation of "peakless" haloes, that cannot be accounted for by any of these methods (Ludlow & Porciani 2011).

Traditional analytical methods have provided foundational insights into the process of halo formation, but they struggle in capturing the full complexity of it. Machine Learning (ML) techniques have emerged as a promising alternative, potentially capable of capturing intricate non-linear dynamics inherent to the gravitational collapse of structures. ML algorithms can be trained on N-body simulations to emulate the results of much more expensive calculations. Previous studies have trained ML models to map initial positions and velocities of particles to their final states (He et al. 2019; Giusarma et al. 2019; Alves de Oliveira et al. 2020; Wu et al. 2021; Jamieson et al. 2022) and to predict the distribution of non-linear density fields (Rodríguez et al. 2018; Perraudin et al. 2019; Schaurecker et al. 2021; Zhang et al. 2023; Schanz et al. 2023).

Further, ML has been used to predict and gain insights into the formation of haloes. Some studies utilized classification methods to

[★] E-mail: daniellopezcano13@gmail.com

anticipate if a particle will become part of a halo (Lucie-Smith et al. 2018; Chacón et al. 2022; Betts et al. 2023), or to predict its final mass category (Lucie-Smith et al. 2019). In Lucie-Smith et al. (2020) a regressor network is trained to predict the final halo mass for the central particle in a given simulation crop. The work by Bernardini et al. (2020) demonstrates how ML-segmentation techniques can be applied to predict halo Lagrangian regions. In Berger & Stein (2019) a semantic segmentation network is trained to predict Peak-Patch-haloes. In Lucie-Smith et al. (2023) a network is trained to predict the mass of haloes when provided with a Lagrangian region centered on the centre-of-mass of proto-halo patches and is then used to study assembly bias when exposed to systematic modifications of the initial conditions.

While interesting qualitative insights have been obtained in these studies, it would be desirable to develop a model that accurately predicts halo membership at a particle level, surpassing some of the limitations from previous works. An effective model should predict particles forming realistic N-body halos, improving upon previous models restricted to simpler halo definitions (e.g. Berger & Stein 2019, where Peak-patch haloes are targeted). Additionally, an ideal model should be able to predict disconnected Lagrangian halo patches, overcoming the limitations of methods like the watershed technique used in Bernardini et al. (2020), which can only handle simply-connected regions. Furthermore, particles within the same halo should share consistent mass predictions, avoiding having different halo mass estimates for particles belonging to the same halo (e.g. Lucie-Smith et al. 2020).

We present a general ML framework to predict the formation of haloes from the initial linear fields. We create a ML model designed to forecast the assignment of individual particles from the initial conditions of an N-body simulation to their respective halos. To do so we train two distinct networks, one for conducting semantic segmentation and another for instance segmentation. These two networks together conform what is known as a panoptic-segmentation model. Our model effectively captures the dynamics of halo formation and offers accurate predictions. We provide the models used in this study for public access through our GitHub repository: https://github.com/daniellopezcano/instance_halos.

The rest of this paper is organized as follows: In §2, we define the problem of identifying different Lagrangian halo regions from the initial density field (§§2.1), introduce the panoptic segmentation method (§§2.2), present the loss function employed to perform instance segmentation (§§2.3), describe the simulations used for model training (§§2.4), asses the level of indetermination for the formation of proto-haloes (§§2.5), outline the CNN architecture (§§2.6), and explain our training process (§§2.7). In §3, we present the outputs of our semantic model (§§3.1) and our instance segmentation approach (§§3.2). We investigate how our model reacts to changes in the initial conditions in §§4.1 & §§4.2, and study how the predictions of our model are affected when varying the cosmology §§4.3. We conclude with a summary and final thoughts in §5.

2 METHODOLOGY

We aim to predict the formation of DM haloes provided an initial density field. To comprehensively address this problem, we divide this section into distinct parts. In §§2.1, we explain the problem of predicting halo-collapse and discuss the most general way to phrase it. In §§2.2, we introduce the panoptic segmentation techniques and explain how they can be employed to predict halo formation. We divide §§2.2 into two separate parts: semantic segmentation and instance segmentation. In §§2.3 we describe the loss function employed to perform instance segmentation. In §§2.4, we present the suite of simulations generated to train and test our models. In §§2.5 we asses the level of indetermination of proto-halo formation. In §§2.6 we explain how to build a high-performance model employing convolutional neural networks. Finally, in §§2.7 we present the technical procedure followed to train our models.

2.1 Predicting structure formation

The goal of this work is to develop a machine learning framework to predict the formation of haloes from the initial conditions of a given universe. Different approaches are possible to define this question in a concrete input/output setting. We want to define the problem in a way that is as general as possible, so that our model can be used in many different contexts.

The input of the model will be the linear density field discretized to a three dimensional grid δ_{ijk}. A slice through such a linear density field is shown in the top panel of Figure 1 and represents how our universe looked at early times, e.g., $z \gtrsim 100$. Beyond the density field, we also provide the linear potential field ϕ_{ijk} as an input. The information included in the potential is in principle degenerate with the density field if the full universe is specified. However, if only a small region is provided, then the potential contains additional information of e.g. the tidal field sourced by perturbations outside of the region considered.

The model shall predict which patches of the initial density field become part of which haloes at later times. Concretely, we want it to group the N^3 initial grid cells (corresponding, e.g., to particles in a simulation) into different sets, so that each set contains exactly all particles that end up in the same halo at a later time. Additionally there has to be one special extra set that contains all remaining particles that do not become part of any halo:

$$\text{Input:} \quad \delta_{ijk}, \phi_{ijk} \tag{1}$$

$$\text{Output:} \quad \overbrace{\{\mathrm{id}_A, \mathrm{id}_B, \ldots\}}^{\text{halo 1}}, \overbrace{\{\mathrm{id}_C, \mathrm{id}_D, \ldots\}}^{\text{halo2}}, \ldots, \overbrace{\{\mathrm{id}_E, \mathrm{id}_F, \ldots\}}^{\text{outside of haloes}}, \tag{2}$$

This task is called in the ML literature an *instance segmentation* problem. Note that it is different from typical classification problems since (A) the number of sets depends on the considered input and (B) the sets have no specific order. In practice, it is useful to define the different sets by assigning different number-labels to them. For example, one possible set of particles belonging to the same halo can be given the label “1”, another set the label “2”, and so forth. These number-labels do not have a quantitative meaning and are permutation invariant, for example, interchanging the label “1” with “2” yields the same sets.

We show such a labelling of the initial space in the bottom panel of Fig. 1. In this case, the labels were inferred by the membership to haloes in an N-body simulation that employs the initial conditions depicted in the top panel of Fig. 1 (see Section 2.4). Our goal is to train a model to learn this instance segmentation into halo sets by training it on the output from N-body simulations.

We note that other studies have characterised the halo-formation processes through a slightly different prediction problem. For example, Lucie-Smith et al. (2020) trains a neural network to predict the final halo masses directly at the voxel-level. While their approach offers insights into halo formation, our method provides a broader perspective: halo masses can be inferred easily through the size of the corresponding sets, but other properties can be inferred as well – for example the Lagrangian shapes of haloes which are important

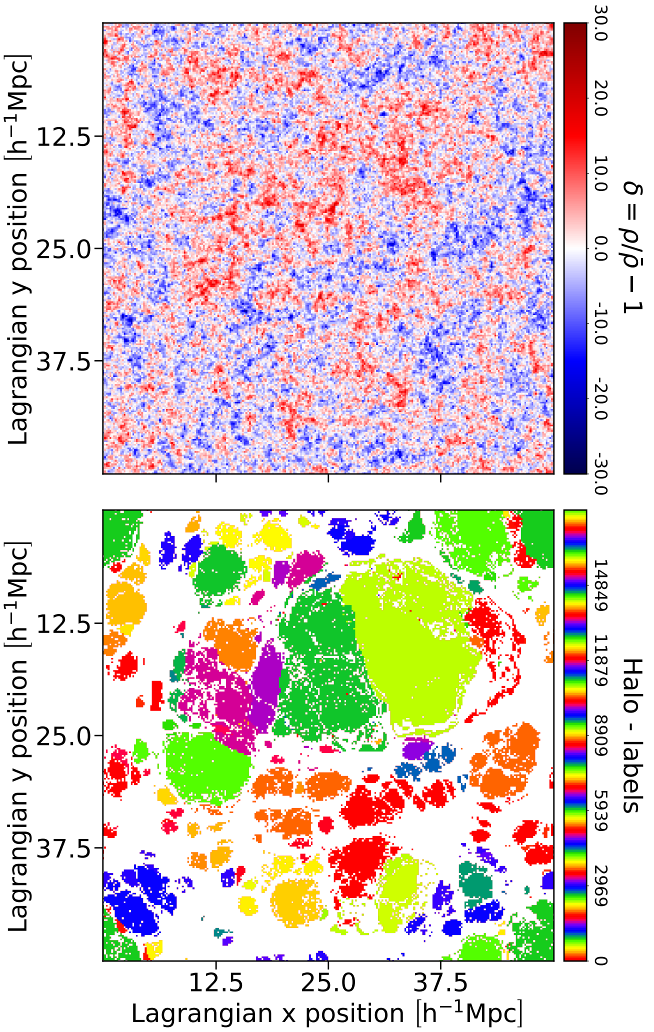

Figure 1. Example of the prediction problem considered in this article. **Top panel**: Slice of the three-dimensional initial density field of an N-body simulation. Each voxel (represented here as a pixel) corresponds to a particle that can become part of a halo at later times. **Bottom panel**: Regions in the initial condition space (same slice as the top panel) that are part of different DM haloes at redshift $z = 0$. Pixels coloured in white do not belong to any halo. Pixels with the same colour belong to the same halo and different colours indicate different haloes. In this work we present a machine learning approach to predict the formation of haloes (as in the bottom panel) from the initial condition field (top panel).

to determine their spin (White 1984). Furthermore, our approach ensures the physical constraint that particles that become part of the same halo are assigned the same halo mass.

2.2 Panoptic Segmentation

The proposed problem requires first to segment the particles semantically into two different classes (halo or non-halo) and then to classify the particles inside the halo class into several different instances. The combination of such semantic plus instance segmentation is sometimes referred to as *panoptic segmentation*. A number of strategies have been proposed to solve such panoptic segmentation problems (Kirillov et al. 2016; Bai & Urtasun 2016; Arnab & Torr 2017; De Brabandere et al. 2017; Kirillov et al. 2018, 2023) and they usually operate in two-steps:

(i) **Semantic segmentation**: The objective of this task is to predict, for each voxel in our initial conditions (representing a tracer particle in the N-body code), whether it will be part of a DM halo at $z = 0$. This task is a classification problem, and we will employ the balanced cross-entropy (BaCE) loss (Xie & Tu 2015) to tackle it:

$$\mathcal{L}_{\mathrm{BaCE}}\left(\mathbf{Y}, \hat{\mathbf{Y}}\right) = -\beta \mathbf{Y} \log \hat{\mathbf{Y}} - (1-\beta)\,(1-\mathbf{Y}) \log(1-\hat{\mathbf{Y}}) \qquad (3)$$

Here, $\mathbf{Y}$ represents the ground truth data vector, each entry corresponds to a voxel and is equal to 1 if the associated particle ends up being part of a halo; otherwise, its value is 0. $\hat{\mathbf{Y}}$ contains the model predictions, with each entry representing the probability that this particle ends up in a halo. The parameter β handles the class imbalance and is calculated as the number of negative samples divided by the total number of samples. We measure β using our training simulations (see §§2.4) and obtain a value of $\beta = 0.5815$.

(ii) **Instance segmentation**: The objective of this task is to recognize individual haloes (instances) by identifying which particles (from those that are predicted to be part of a DM halo) belong to the same object and separating them from others.

Instance segmentation tasks are not conventional classification problems and tackle the problems of having a varying number of instances and a permutational-invariant labelling. To our knowledge there is no straightforward way to phrase the problem of classifying each voxel into a flexible number of permutable sets through a differentiable loss function. Typical approaches train a model to predict a related differentiable loss and then apply a postprocessing step on top of it. Unfortunately, this leads to the loss function not directly reflecting the true objective.

Various approaches have been proposed to tackle this problem (Kirillov et al. 2016; Bai & Urtasun 2016; Arnab & Torr 2017; De Brabandere et al. 2017; Kirillov et al. 2018, 2023). A popular method is the watershed technique (Kirillov et al. 2016; Bai & Urtasun 2016). This method uses a network to predict a semantic segmentation and the borders of different instances (Deng et al. 2018) and then applies a watershed algorithm to separate different instances in a post-processing step. However, the watershed approach comes with a number of limitations:

- It cannot handle the identification of disconnected regions belonging to the same instance, a problem known as occlusion.
- It is necessary to select appropriate threshold values for the watershed post-processing step to generate the final instance map. These parameters are typically manually chosen to match some particular metric of interest, but might negatively impact the prediction of other properties. For instance, in Bernardini et al. (2020), they apply the watershed technique to predict Lagrangian halo regions identified with the HOP algorithm (Eisenstein & Hut 1998). However, they choose the watershed threshold to reproduce the halo-mass-function, which does not ensure that the Lagrangian halo regions are correctly predicted.
- The watershed approach would struggle to identify the borders of Lagrangian halo regions since they are difficult to define. In Fig. 3 it can be appreciated that the borders of halo regions are very irregular. There also exist points in the "interior" of these regions which are "missing" and make it particularly complex to define the border of a halo.

Despite all the challenges presented by the watershed approach, in §A, we apply this method to predict the formation of FoF-haloes and discuss how the border-prediction problem can be addressed.

An approach that offers greater flexibility for grouping arbitrarily arranged particles was presented by De Brabandere et al. (2017). We will follow this approach through the remainder of this work. The main idea behind this method, which we will refer to as the "Weinberger approach"[1], is to train a model to produce a "pseudo-space representation" for all the elements of our input space (i.e., voxels/particles in the initial conditions). An ideal model would map voxels belonging to the same instance close together in the pseudo-space while separating them from voxels belonging to different instances. Consequently, the pseudo-space distribution would consist of distinct clouds of points, each representing a different instance (see Fig. 2). The postprocessing step required to generate the final instance segmentation in the Weinberger approach is a clustering algorithm which operates on the pseudo-space distributions.

2.3 Weinberger loss

The Weinberger approach possesses some advantages over other instance segmentation techniques: First of all, the loss function more closely reflects the instance segmentation objective; that is, to classify different instances into a variable number of sets which are permutationally-invariant. Secondly, the approach is more flexible and makes less assumptions, for example, it can handle occlusion cases and does not need to assume the existence of well defined instance borders. Finally, the Weinberger loss hyperparameters δ_{Pull} and δ_{Push} define a natural clustering scale which reduces the necessity to manually over-tune the postprocessing step to obtain the final instance maps (see De Brabandere et al. 2017, and references therein).

In Fig. 2, we schematically illustrate the effects of the individual components of the Weinberger loss. Each point in this figure represents a pseudo-space embedding of an input voxel. The colors indicate the assigned labels based on the ground truth. Points sharing the same color belong to the same instance (according to the ground truth), whereas different colors depict separate instances. The "centre of mass" for each cluster is computed and indicated with colored crosses as "cluster centres". The Weinberger loss is constituted by three separate terms:

- **Pull force,** equation (4):

$$L_{pull} = \frac{1}{C}\sum_{c=1}^{C}\frac{1}{N_c}\sum_{i=1}^{N_c}\max\left((\|\mu_{\mathbf{c}} - \mathbf{x_i}\| - \delta_{\text{Pull}})^2, 0\right) \quad (4)$$

Given a certain instance c (where C is the total number of instances), a point i belonging to that set, whose pseudo-space position is $\mathbf{x_i}$, will feel an attraction force proportional to the distance to the instance centre $\mu_c = \sum_{i=1}^{N_c}\mathbf{x_i}/N_c$, where N_c is the number of members associated with the instance c. Points closer than δ_{Pull} (which is a hyperparameter of the Weinberger loss) from the instance center will not experience any pull force. The pull force is represented in Fig. 2 as coloured arrows pointing towards the instance centres outside the solid-line circles, which symbolize the distance δ_{Pull} to the instance centres.

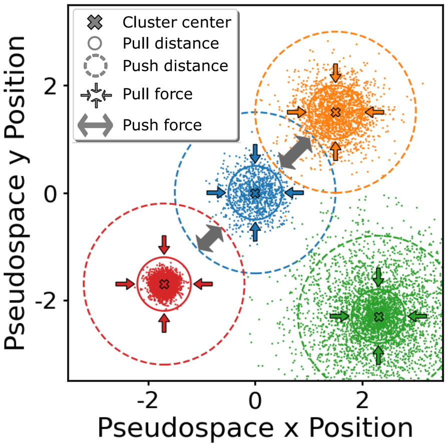

Figure 2. Example of a two-dimensional pseudo-space employed to separate different instances according to the Weinberger loss. Coloured points represent individual points mapped into the pseudo-space. The centres of the clusters are presented as coloured crosses. Coloured arrows depict the influence of the pull force term, only affecting points outside the δ_{Pull} range of their corresponding cluster centre. Grey arrows show the influence of the push force that manifest if two cluster centres are closer than the distance $2 \cdot \delta_{\text{Push}}$

- **Push force,** equation (5):

$$L_{\text{push}} = \frac{1}{C(C-1)}\sum_{\substack{c_A=1\\ c_A\neq c_B}}^{C}\sum_{c_B=1}^{C}\max\left(\left(2\delta_{\text{Push}} - \left\|\mu_{\mathbf{c_A}} - \mu_{\mathbf{c_B}}\right\|\right)^2, 0\right) \quad (5)$$

Two instances A and B will repel each other if the distance between their instance centres in the pseudo-space, μ_{c_A} and μ_{c_B}, is smaller than $2\delta_{\text{Push}}$ (a hyperparameter of the Weinberger loss). The force they feel is proportional to the distance between them. In Fig. 2 the push force is represented as grey arrows. The dashed circles represent the distance δ_{Push} to the instance centres.

- **Regularization force,** equation (6):

$$L_{\text{reg}} = \frac{1}{C}\sum_{c=1}^{C}\|\mu_{\mathbf{c}}\| \quad (6)$$

To avoid having an arbitrarily big pseudo-space distribution all instance centers will feel an attraction towards the pseudo-space origin.

The overall effect of these forces on the total Weinberger loss is written as:

$$\mathcal{L}_{\text{Wein}} = c_{\text{Pull}} \cdot L_{\text{Pull}} + c_{\text{Push}} \cdot L_{\text{Push}} + c_{\text{Reg}} \cdot L_{\text{Reg}} \quad (7)$$

Where c_{Pull}, c_{Push}, and c_{Reg} are hyperparameters that regulate the strength of the different components.

Minimizing equation (7) ensures that the pseudo-space mapping produces instance clusters separated from each other. A model trained effectively will predict pseudo-space distributions with points corresponding to the same instances being grouped together and distinctly separated from other instances. In an ideal scenario in which the Weinberger loss is zero, all points are closer than δ_{Pull} to their corresponding cluster centers, and clusters are at least $2\delta_{\text{Push}}$ apart. However, realistically, the Weinberger loss won't be exactly zero, necessitating a robust clustering algorithm for accurate instance map predictions.

[1] The loss function employed by De Brabandere et al. (2017) to perform instance segmentation is inspired by a loss function originally proposed by Weinberger & Saul (2009) in the context of contrastive learning as a triplet-loss function.

In Appendix B we describe the clustering algorithm that we have developed to robustly identify the different instance maps. In our clustering algorithm we first compute the local density for each point in our pseudo-space based on a nearest neighbors calculation. We then identify groups as descending manifolds of density maxima surpassing a specified persistence ratio threshold. Particles are assigned to groups according to proximity and density characteristics. We merge groups selectively, ensuring that the persistence threshold is met. The algorithm relies on three key hyper-parameters for optimal performance: $N_{\rm dens}$, $N_{\rm ngb}$ and $p_{\rm thresh}$. This approach effectively segments the pseudo-space distribution of points, even when perfect separation is not achieved, thus enhancing the reliability for predicted instance maps.

2.4 Dataset of Simulations

We generate twenty N-body simulations with different initial conditions to use as training and validations sets for our panoptic segmentation model. Our simulations are carried out using a lean version of `L-Gadget3` (see Springel et al. 2008; Angulo et al. 2012, 2021). For each of these simulations, we evolve the DM density field employing $\mathrm{N_{DM}} = 256^3$ DM particles in a volume of $V_{\rm box} = (50\,h^{-1}\mathrm{Mpc})^3$, resulting in a DM particle-mass of $m_{\rm DM} = 6.35 \cdot 10^8\,h^{-1}\mathrm{M}_\odot$. All our simulations employ the same softening length: $\epsilon = 5\,h^{-1}$kpc, and share the cosmological parameters derived by Planck Collaboration et al. (2020), that is, $\sigma_8 = 0.8288$, $n_{\rm s} = 0.9611$, $h = 0.6777$, $\Omega_{\rm b} = 0.048252$, $\Omega_{\rm m} = 0.307112$, and $\Omega_\Lambda = 0.692888$. Our suite of simulations is similar to the one employed in Lucie-Smith et al. (2020).

We use a version of the `NgenIC` code (Springel 2015) that uses second-order Lagrangian Perturbation Theory (2LPT) to generate the initial conditions at $z = 49$. We employ a different random seed for each simulation to sample the Gaussian random field that determines the initial density field. We identify haloes at redshift $z = 0$ in our simulations using a Friends-of-Friends algorithm (Davis et al. 1985), with linking length $b = 0.2$. In this work, we will only consider haloes formed by 155 particles or more, corresponding to $M_{\rm FoF} \gtrsim 10^{11}\,h^{-1}\mathrm{M}_\odot$. We use 18 of these simulations to train our model and keep 2 of them to validate our results.

2.5 Assessing the level of indetermination

In addition to the training and test sets, we run a set of simulations to establish a target accuracy for our model. These simulations test to what degree small sub-resolution changes of the initial density field can affect the final Lagrangian halo regions.

Structure formation simulations resolve the initial conditions of a considered universe only to a limited degree and exhibit therefore an inherent degree of uncertainty. (1) The numerical precision of simulations is limited (e.g. to 32bit floating point numbers) and therefore any results that depend on the initial conditions beyond machine-precision are inherently uncertain. For example, Genel et al. (2019) show that changes in the initial displacement of N-body particles at the machine-precision level can lead to differences in the final locations of particles as large as individual haloes. (2) The initial discretization can only resolve the random perturbations of the Gaussian random field down to a minimum length scale of the mean-particle separation. If the resolution of a simulation is increased, then additional modes enter the resolved regime and act as additional random perturbations. Such additional perturbations may induce some random changes in the halo assignment of much larger scale structures.

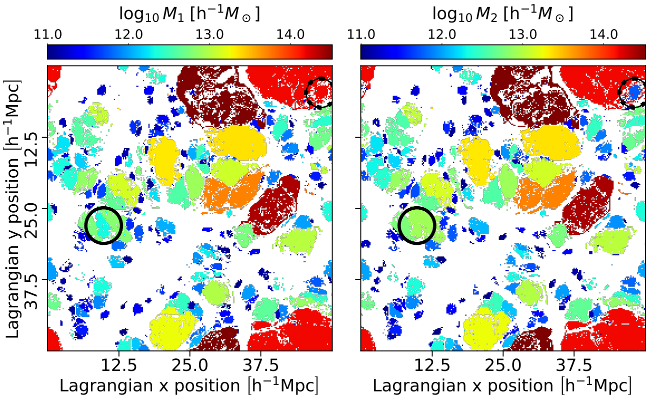

Figure 3. Slice of the Lagrangian halo regions of the two "baseline" simulations (left and right panels respectively). These simulations only differ in sub-resolution perturbations to the initial conditions and their level of agreement sets a baseline for the desired accuracy of our models. The colors employed for both panels represent the mass of the halo associated with each particle for the different Lagrangian halo patches. Circled regions highlight Lagrangian patches whose associated mass significantly changes between the two simulations.

A good model should learn all aspects of structure formation that are certain and well resolved at the considered discretization level. However, there is little use in predicting aspects that are underspecified and may change with resolution levels. Therefore, we conduct an experiment to establish a baseline of how accurate our model shall be.

We run two additional $N = 256^3$ simulations with initial conditions generated by `MUSIC` code (Hahn & Abel 2011). For these simulations we keep all resolved modes fixed (up to the Nyquist frequency of the 256^3 grid), but we add to the particles different realisations of perturbations that would be induced by the next higher resolution level. We do this by selecting every 2^3th particle from two initial condition files with 512^3 particles and with different seeds at the highest level ("level 9" in `MUSIC`). Therefore, the two simulations differ only in the random choice of perturbations that are unresolved at the 256^3 level. We refer to these two simulations as the "baseline" simulations.

In Fig. 3 we show a slice of the Lagrangian halo patches at $z = 0$ through these simulations (left and right panels respectively). The color-map in this Figure represent the masses of the halo that each particle become part of, which correspond to the size of the corresponding halo-set. We color each pixel (which corresponds to a certain particle) according to the mass of the halo that it belongs to. We can appreciate that the outermost regions of the Lagrangian regions are particularly affected while the innermost parts remain unchanged. Notably, in certain instances, significant changes appear due to the merging of haloes in one of the simulations where separate haloes are formed in the other (black circled regions).

Throughout this article we will use the degree of correspondence between the baseline simulations as a reference accuracy level. We consider a model close to optimal if the difference between its predictions and the ground truth are similar to the differences observed between the two baseline simulations. A lower accuracy than this would mean that a model has not optimally exploited all the information that is encoded in the initial conditions. A higher accuracy than this level is not desirable, since it is not useful to predict features that

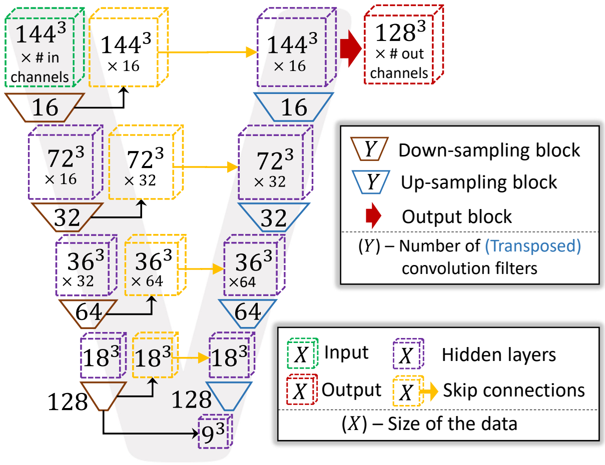

Figure 4. Flowchart of the particular V-Net architecture we have implemented. The network can take as input multiple channels with dimensions of 144^3 (top left green cube) and generates predictions for the central voxels with dimensions 128^3 (top right red cube). The flowchart illustrates the encoder and decoder paths, along with other distinctive features of the network. Notably, the hidden layers and skip connections are represented by purple and yellow cubes, with their respective dimensions annotated at their centres. The down-sampling and up-sampling blocks are shown as brown and purple trapezoids, in their centres we indicate the number of filters employed for the convolution (or transposed convolution) operations.

depend on unresolved aspects of the simulation and may be changed by increasing the resolution level.

2.6 V-Net Architecture

V-nets are state-of-the-art models, product of many advances in the field of ML over the last decades (Fukushima 1980; Lecun et al. 1998; Krizhevsky et al. 2012; Szegedy et al. 2014; Long et al. 2014; Ronneberger et al. 2015; He et al. 2015). They are a particular kind of convolutional neural network (CNN) developed and optimized to efficiently map between volumetric inputs and volumetric outputs. V-nets are formed by two separate modules: the encoder (or contracting path) which learns how to extract large-scale abstract features from the input data; and the decoder (or up-sampling path) that translates the information captured by the encoder to voxel-level predictions (also making use of the information retained in the "skipped connections"). We train V-nets to minimize the loss functions presented in §§2.2 and §§2.3. We now explain the technical characteristics of how we have implemented a V-net architecture in TensorFlow (Abadi et al. 2015) (see Fig. 4 for a schematic representation of our network architecture):

- Input: Our network is designed to accept as input 3D crops consisting of 144^3 voxels.[2] For the results presented in §3, we employ two input channels for the semantic segmentation model, corresponding to the initial density field and the displacement potential, which is defined through Poisson's equation as:

$$\delta(\vec{q}) = \vec{\nabla}^2\phi(\vec{q}) \tag{8}$$

For the instance segmentation model, we include three additional input channels corresponding to the Lagrangian positions of particles. This is necessary, since the network has to be able to map different haloes with the same density (and potential) structure at different locations in the initial field to different locations in the pseudo space.

- Encoder / contractive / down-sampling / down-scaling path: This module consists of consecutive down-scaling blocks that reduce the number of voxels per dimension by half at each level of the network. The purpose of the down-scaling path is to enlarge the network's field of view, enabling per-voxel predictions that take into account distant regions of the field. Achieving this would be impractical using large convolution kernels, as they would consume excessive memory. Within each down-sampling block, we apply three consecutive convolution operations followed by a Leaky-ReLu activation function. The number of convolution filters in a contractive block doubles with each level of compression to improve the performance of the model. For each level, the latent maps computed before the final convolution (the one used to reduce the data size) are temporarily stored to serve as a skip connection for the up-scaling path. In Fig. 4 we show the dimensions of the latent maps computed at each level of the contractive path; the deepest level of our network has a size of $9^3 \times 128$.
- Decoder / up-sampling / up-scaling path: This path operates opposite to the contractive path; each up-scaling block doubles the number of voxels per dimension, ultimately recovering an image with the same dimensions as the original input (see Fig. 4). The up-sampling path facilitates the extraction of smaller-scale features that influence the final per-voxel predictions. Within an up-sampling block, the final convolution is substituted with a transposed convolution operation, that allows doubling the output size per dimension.
- Output: The final module of our network takes as input the latent maps with dimensions $144^3 \times 16$. The functionality of this module varies depending on the task at hand. For semantic segmentation, a single convolution operation is performed, resulting in a latent map of $144^3 \times 1$. This map is subsequently cropped to $128^3 \times 1$, and finally, a sigmoid activation function is applied. In the case of instance segmentation, we have decided to work in a three-dimensional pseudo-space, hence, we employ a convolution with three filters to obtain $144^3 \times 3$ maps, which are afterwards cropped to $128^3 \times 3$. In both cases, the final cropping operation is implemented to enhance model performance by focusing on the central region of the image.

The V-Net architecture we have implemented is a state-of-the-art model that encompasses over $3 \cdot 10^6$ trainable parameters.

2.7 Training

We train our segmentation networks using a single Nvidia Quadro RTX 8000 GPU card. As mentioned in §§2.4, we employ 18 simulations for training, dividing the training process into separate stages for the semantic and instance models.

To ensure robust training and enhance the diversity of training examples without needing to run more computationally expensive simulations, we apply the following data augmentation operations each time we extract a training sample from our simulation suite:

(i) Select one of the training simulation boxes at random.
(ii) Select a random voxel as the center of the input/output regions.
(iii) Extract the input (144^3) and target (128^3) fields of interest by cropping the regions around the central point, considering the periodic boundary conditions of the simulations.

[2] Ideally, we would prefer to accept as input 256^3 voxels (corresponding to the full simulation box). However, our GPU resources, though powerful (specifically, an `NVIDIA QUADRO RTX 8000` with 48 GB of memory), are insufficient to accommodate such an input size while maintaining a reasonably complex network architecture.

Table 1. Hyper-parameters employed in our instance segmentation pipeline.

δ_{Pull}	δ_{Push}	c_{Pull}	c_{Push}	c_{Reg}	N_{dens}	N_{ngb}	p_{thresh}
0.5	1.5	1	1	0.001	20	15	4.2

(iv) Randomly transpose the order of the three input grid dimensions q_x, q_y, q_z.

(v) Randomly chose to flip the axes of the input fields.

To train our semantic and instance segmentation networks we minimize the respective loss functions – equation (3) and equation (7) – employing the Adam optimizer implemented in `TensorFlow` (Abadi et al. 2015). We train our models for over 80 epochs, each epoch performs mini-batch gradient descent using 100 batches, and each batch is formed by 2 draws from the training simulations. We deliberately choose a small batch size to avoid memory issues and ensure the network's capability to handle large input and output images (144^3 and 128^3 respectively). Selecting a small batch size induces more instability during training; we mitigate this issue by using the clip normalization operation defined in `TensorFlow` during the backpropagition step.

The hyper-parameter β in the Balanced Cross-Entropy equation (3) is determined by computing the ratio of negative samples to the total number of samples in the training data. The value of β measured in different training simulations lays in the interval $[0.575, 0.5892]$. There exists a slight predominance of voxels/particles that do not collapse into DM haloes with mass $M_{\mathrm{FoF}} \gtrapprox 10^{11}\,h^{-1}\mathrm{M}_{\odot}$ at $z = 0$ considering the Planck Collaboration et al. (2020) cosmology. We fix the hyper-parameter β in equation (3) to the mean value $\beta = 0.5815$.

Regarding the hyper-parameters in the Weinberger loss equation (7), we adopt the values presented in De Brabandere et al. (2017), as we have observed that varying these parameters does not significantly affect our final results. The specific hyper-parameter values are the following: $c_{\mathrm{Pull}} = 1$, $\delta_{\mathrm{Pull}} = 0.5$, $c_{\mathrm{Push}} = 1$, $\delta_{\mathrm{Push}} = 1.5$, and $c_{\mathrm{Reg}} = 0.001$. We have conducted a hyper-parameter optimization for the clustering algorithm described in Appendix B and found the following values: $N_{\mathrm{dens}} = 20$, $N_{\mathrm{ngb}} = 15$ and $p_{\mathrm{thresh}} = 4.2$ (see Table 2).

3 MODEL EVALUATION

In this section, we test the performance of our models for semantic segmentation (§§3.1) and instance segmentation (§§3.2). We use the two simulations reserved for validation to generate the results presented in this section.

3.1 Semantic Results

In Fig. 5, we compare the predictions of the semantic segmentation network with the halo segmentation found in the validation simulation. The leftmost panel illustrates a slice of the ground truth. Voxels/particles of the initial conditions belonging to a DM halo at $z = 0$ are shown in red; blue voxels represent particles not belonging to a DM halo at $z = 0$.

The central panel of Fig. 5 displays the probabilistic predictions from our semantic model for the same slice. The colour map indicates the probability assigned to each pixel for belonging or not to a DM halo. Voxels with a white colour have a 50% predicted probability of belonging to a halo. The neural network exhibits a tendency to smooth out features, assigning uncertain probabilities to regions near halo borders, while consistently assigning high probabilities to inner regions and low probabilities to external regions. This smoothing effect causes the predicted regions to not present any "gaps". The gaps in the Lagrangian shapes of haloes seem to be a feature very sensitive to the initial conditions and impossible to capture accurately at a voxel level. This is supported by the fact that the gaps also change significantly in the baseline simulations (see Fig. 3).

The rightmost panel of Fig. 5 shows the pixel-level error map for the same slice. We select a semantic threshold value equal to 0.589 to generate these results. We choose this value for the semantic threshold so that the total predicted number of particles that belong to a halo matches the number collapsed voxels in the validation simulations. In Appendix C we further analyze the sensitivity of our semantic results to the value chosen for the semantic threshold. We use different colours to represent the corresponding classes of the confusion matrix: Green corresponds to true positive (TP) cases, blue to true negatives (TN), black to false negatives (FN), and red to false positives (FP).

Some regions are particularly challenging to predict for the network, likely due to their sensitivity to changes in the initial conditions. For example, in the rightmost panel of Fig. 5, it is easy to appreciate many FN regions that appear as black string-like structures surrounding TP collapsed regions. These FN cases likely correspond to particles infalling into the halo at $z = 0$, identified as part of the FoF group despite not having completed the first pericentric passage. Capturing this behaviour might be particularly challenging for the network since the exact shape of these "first-infall" regions is more sensitive to small changes in the initial conditions. Also, we can appreciate FP regions that appear between the FN string-like regions and the TPs corresponding to the central Lagrangian regions of haloes. Additionally, the boundaries of the largest haloes may be especially difficult to predict for the network, since they only fit partially into the field of view.

The results presented in Fig. 5 suggest, upon visual inspection, that our model accurately captures many of the complex dynamics that determine halo collapse. To rigorously assess the performance of our model we need to quantify the results obtained from our semantic network and compare them with the differences between the baseline simulations, as discussed in §2.

In Table 2 we present the values of some relevant metrics that we can employ to evaluate the performance of our semantic network (we have considered the semantic threshold of 0.589). In particular we study the behaviour of five different metrics: True Positive Rate $\mathrm{TPR} = \mathrm{TP}/(\mathrm{TP}+\mathrm{FN})$, True Negative Rate $\mathrm{TNR} = \mathrm{TN}/(\mathrm{TN}+\mathrm{FP})$, Positive Predictive Value $\mathrm{PPV} = \mathrm{TP}/(\mathrm{TP}+\mathrm{FP})$, Accuracy ACC and the F_1-score (which is a more representative score than the accuracy when considering unbalanced datasets), see equation (9):

$$\mathrm{ACC} = \frac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FN}+\mathrm{FP}} \quad ; \quad \mathrm{F}_1 = \frac{2\mathrm{TP}}{2\mathrm{TP}+\mathrm{FP}+\mathrm{FN}} \tag{9}$$

Table 2 also contains the scores measured using the baseline simulations. Our model returns values for all the metrics very close to the optimal target from the baseline simulations. This demonstrates the reliability of our model in predicting the well-specified aspects of halo collapse. See Appendix C for a more detailed discussion about the performance of our semantic model and the relation between the selected semantic threshold with the results contained in Table 2.

In Fig. 6 we compare the values of the predicted TPR as a function of ground truth halo mass ($\mathrm{TPR}_{\mathrm{Pred}}$, solid green line), with the TPR values measured from the baseline simulations ($\mathrm{TPR}_{\mathrm{base}}$, solid black line). It is possible to perform this comparison for the TPR because, in the ground truth data, we retain information about the mass of

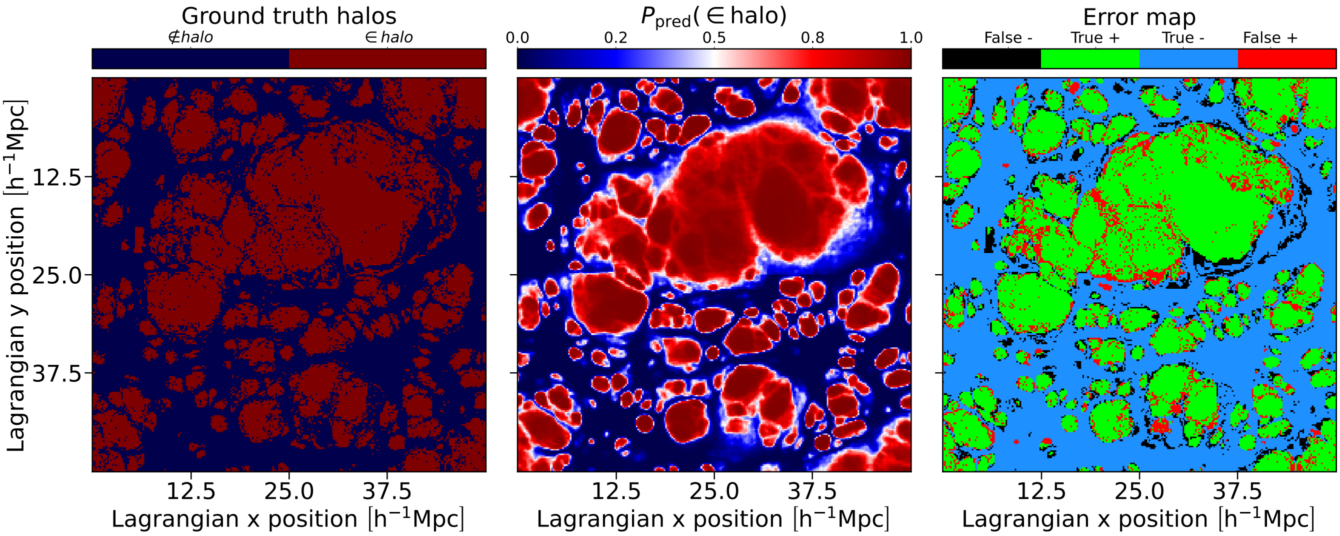

Figure 5. Slice through the predictions of our semantic segmentation network applied to a validation simulation. **Left panel**: Ground truth representation showing in red the voxels/particles belonging to a DM halo at $z = 0$ and in blue those particles that do not belong to a DM halo. **Central panel**: Probabilistic predictions of the semantic network with colour-coded probabilities for halo membership. **Right panel**: Pixel-level error map indicating true positive (green), true negative (blue), false negative (black), and false positive (red) regions resulting after applying a semantic threshold of 0.589 to our predicted map. The network effectively captures complex halo boundaries and exhibits high validation accuracy (acc = 0.86) and F_1-score (F_1 = 0.83).

Table 2. Performance metrics of our semantic segmentation model and comparison with the optimal target accuracy estimated from the baseline simulations. The table presents True Positive Rate (TPR), True Negative Rate (TNR), Positive Predictive Value (PPV), and Negative Predictive Value (NPV).

	TPR	TNR	PPV	ACC	F_1
Pred.	0.838	0.883	0.838	0.864	0.838
Optimal	0.887	0.914	0.882	0.903	0.884

the FoF-haloes associated with each DM particle. Therefore, we can compute the fraction of TP cases in different ground-truth-mass-bins by selecting the voxels according to the mass associated with them in the ground truth.

In Fig. 6, the values for $\rm TPR_{base}$ increase with halo mass, indicating that particles that end up in lower-mass haloes are more sensitive to small-scale changes in the initial conditions, consequently, harder to predict accurately. Our network's predictions follow a similar trend, albeit with some discrepancies. The model seems to under-predict the number of particles that end up in haloes with masses lower than $M_{\rm True} \lessapprox 10^{12}\,h^{-1}{\rm M}_\odot$ (dotted vertical black line in Fig. 6). This indicates that our model tends to under-predict the number of pixels that are identified as TPs in the lower mass end. For haloes whose mass is greater than $10^{12}\,h^{-1}{\rm M}_\odot$, our model returns accurate predictions to a good degree over a broad range, extending more than two orders of magnitude in halo mass.

In this subsection we have demonstrated that our semantic model extracts most of the predictable aspects of halo formation by comparing our results with the baseline simulations (which only differ in unresolved aspects of the initial conditions). We now employ the predictions of our semantic network to generate the final results using our instance segmentation model.

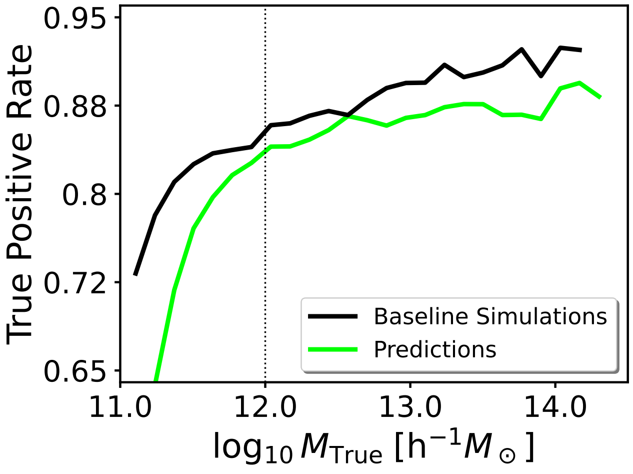

Figure 6. True Positive Rate expressed as a function of the halo mass associated with the ground truth voxels. We present the results measured from the model predictions (solid bright green line) in comparison to the optimal target accuracy from the baseline simulations (solid black line). The vertical dotted line at $10^{12}\,h^{-1}{\rm M}_\odot$ marks the point where model predictions start to differ from the baseline results.

3.2 Instance Results

We want to generate instance-map predictions analogous to the bottom panel of Fig. 1. For this purpose, it is necessary to consider the semantic results. We only compute the instances associated with particles/voxels predicted to be members of haloes by the semantic model. The particles that are predicted to be part of the background are excluded from the pseudo-space representations which are used to determine the different halo-instances.

As discussed in §§2.6, our instance pipeline generates crops with

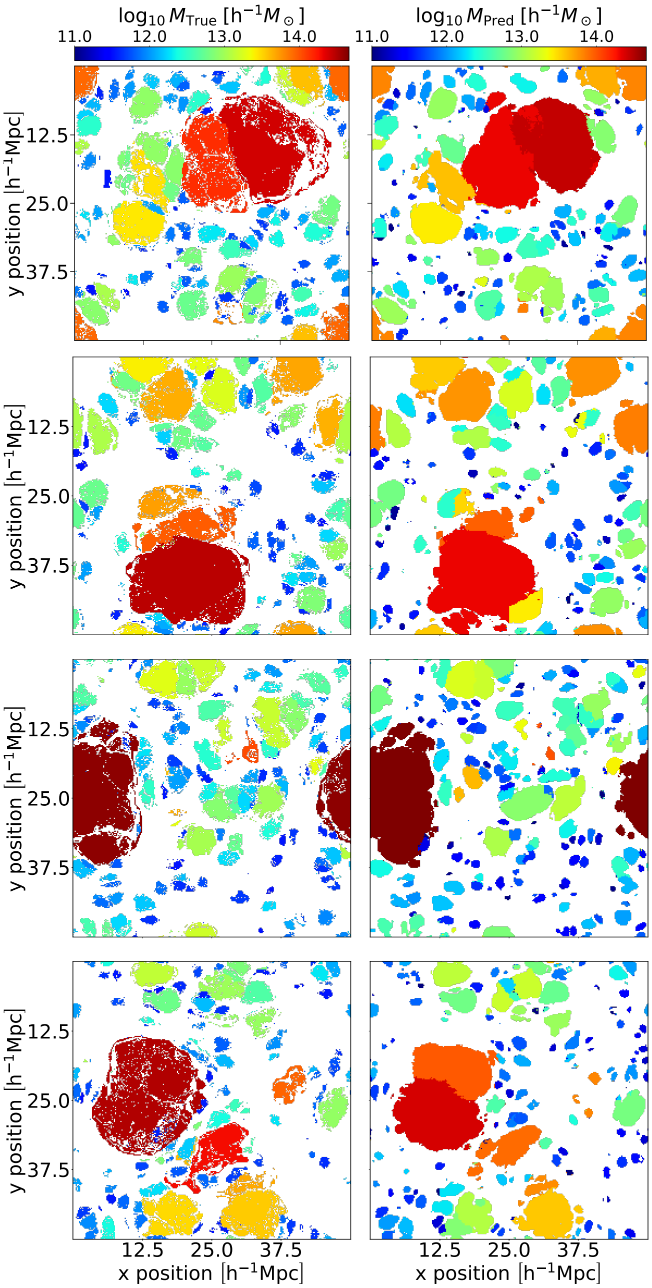

Figure 7. Examples of the instance segmentation results obtained with our model. **Left column**: ground truth masses obtained using N-body simulations. **Right column**: predicted masses obtained using our instance segmentation pipeline. The model is able to predict the Lagrangian patches of haloes, although some small differences – e.g. regarding the connectivity of haloes – exist.

dimensions of 128^3. This constraint arises from technical limitations regarding GPU memory. However, our validation boxes have 256^3 volumes and possible applications may require even much larger boxes. Consequently, we need to develop an algorithm that effectively combines different sub-volume-cropped predictions to generate the final full-box map. The detailed algorithm for combining different instance crops is presented in Appendix D. In summary, we generate two overlapping lattices, the first one is constituted by a set of independent sub-volumes that cover the entire simulation box; the second lattice is centred on the nodes of the first one and also covers the whole simulation box. The overlapping regions between these two lattices is employed to determine whether instances identified in the different crops should merge together or not. We have checked that this procedure does not significantly impact the performance of our network.

We provide some examples of our instance predictions in Fig. 7. The left column displays the ground truth masses of halo Lagrangian regions extracted from the simulation results (analogous to Fig. 3); the right column shows the predictions obtained from our segmentation pipeline. The way in which we compute halo masses from the instance predictions is by counting the number of particles/voxels that have been assigned to the same label and multiplying that by the particle mass of our simulations, $m_{\rm DM} = 6.35 \cdot 10^8\ h^{-1}{\rm M}_\odot$.

The shapes of the halo contours are well captured thanks in part to the semantic predictions. The instance segmentation pipeline successfully distinguishes the different haloes that have formed, and in most cases, correctly separates neighbouring haloes. This is not a trivial task since the size of halo Lagrangian regions varies across several orders of magnitude. Therefore, the instance segmentation pipeline needs to correctly separate wildly different particle groupings in the pseudo-space. Fig. 7 shows that our instance segmentation pipeline correctly identifies different Lagrangian halo regions for the majority of cases. However, we note that differences arise on the one hand for very small haloes that are close to the resolution limit and on the other hand for very large haloes that are larger than the field of view of the network.

In Fig. 8, we present a comparison between the ground truth halo masses and the predicted masses associated with the particles/voxels in our validation set. To generate these results we apply the following procedure: We select all the ground truth voxels/particles that end up in FoF-haloes and study the predictions associated with them. For all the voxels predicted to belong to a DM halo, we can associate a predicted mass. In these cases, we can compare the predicted mass values ($M_{\rm Pred}$) with the ground truth masses ($M_{\rm True}$) at a voxel level. This comparison is shown in the main panel of Fig. 8 as black violin plots (“violins” henceforth). The mass range covered by the black violins goes from $M_{\rm True} = 10^{11}\ h^{-1}{\rm M}_\odot$, corresponding to the minimum mass of haloes (155 particles), to $M_{\rm True} \approx 10^{14.7}\ h^{-1}{\rm M}_\odot$, which is the mass of the most massive halo identified in the validation simulations. The number of high-mass haloes is smaller than small-mass ones and therefore the higher-mass end of the violin plot exhibits more noise. We can appreciate that the median predictions (black dots) correctly reproduce the expected behaviour (ground truth) for around several orders of magnitude.

The voxels identified as part of a halo in the ground truth, but not in the predicted map, are false negative (FN) cases. For these occurrences we can study the dependence of the False Negative Rate (FNR) as a function of the ground truth halo mass (solid black line on the top panel of Fig. 8; analogous to 6). We can also study the reciprocal case in which a voxel is predicted to be part of a halo (hence, it has an associated $M_{\rm Pred}$) but the ground truth voxel is not collapsed. These cases correspond to False positives (FP) but to make a comparison as a function of mass we can only express it in terms of the predicted mass. Therefore, we show as a dashed black line in the top panel of Fig. 8 the false discovery rate,

$$\mathrm{FDR} = \frac{[\mathrm{FP}|M_{\rm Pred}]}{[\mathrm{TP}|M_{\rm Pred}] + [\mathrm{FP}|M_{\rm Pred}]} \quad . \tag{10}$$

We compare our results with those obtained from the baseline simulations. In the main panel of Fig. 8 we present the corresponding

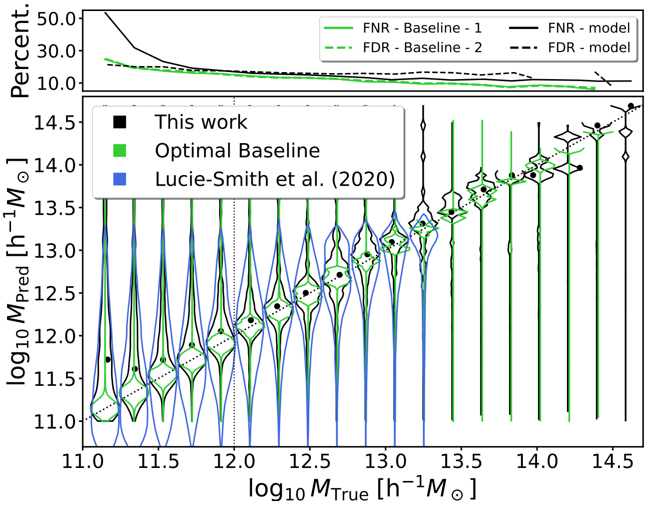

Figure 8. "Violin plot", visualizing the distribution of predicted halo masses (at a voxel level) for different ground-truth mass bins. The black violin plots show the results obtained with our instance segmentation model. Green violin plots show the agreement between the two baseline simulations – representing an optimal target accuracy. The blue violin plots in the main panel show the results presented in Lucie-Smith et al. (2020). The solid black line in the top panel shows the false negative rate, FNR, as a function of the ground truth halo mass. The dashed black line represents the fraction of predicted collapsed pixels that are not actually collapsed as a function of predicted halo mass (false discovery rate, FDR). The green lines on the top panel correspond to the analogous results obtained from the baseline simulations. The model predicts haloes accurately object-by-object for masses $M \gtrsim 10^{12} M_\odot/h$.

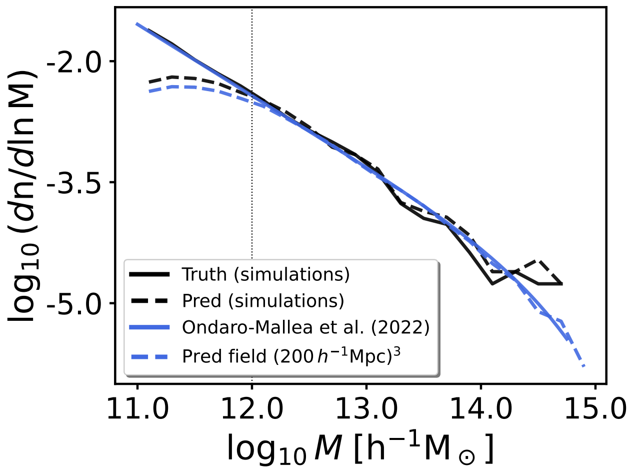

Figure 9. Halo-mass-function (HMF) computed using our N-body simulations reserved for validation (solid black line). The dashed black line represents the predicted HMF using the Lagrangian halo regions obtained with our instance segmentation pipeline. The solid blue line shows the HMF prediction from Ondaro-Mallea et al. (2022). The dashed blue line corresponds to the HMF obtained after evaluating our model in a simulation with 1024^3 particles and $V_{\rm box} = (200\,h^{-1}{\rm Mpc})^3$.

violin plots from the baseline simulations with green lines. The range that the green violins span is smaller than the black violins since the most massive halo identified in the baseline simulations has a mass of $M_{\rm True} \approx 10^{14.4}\,h^{-1}{\rm M}_\odot$. In the top panel, the solid and dashed green lines represent the FPR and FDR respectively. As expected, the FPR and FDR coincide in the case of the baseline simulations. The top panel results demonstrate that our predictions are comparable to those of the baseline simulations (as pointed out in Fig. 6) over most of the considered mass-range. However, they get progressively worse for masses below $M_{\rm True} \lessapprox 10^{12}\,h^{-1}{\rm M}_\odot$ (vertical dotted black line), deviating from the baseline trend. This indicates that our model struggles to capture the correct behaviour of lower mass haloes but it produces accurate predictions for higher-mass ones. When comparing the violin plot distributions of our model with the baseline simulations we appreciate that we obtain similar (but slightly broader) contours. Being able to achieve a similar scatter as in the baseline simulations indicates that our model is able to capture the well-resolved aspects of halo formation. We want to emphasize that precise predictions for halo masses are not directly enforced through the training loss, but are a side product, consequence of precisely reproducing halo Lagrangian patches. The scatter broadens for smaller halo mass and the network looses accuracy in this cases, sometimes associating smaller haloes close to a big Lagrangian patch to its closest more-massive neighbour.

In the main panel of Fig. 8 we include the violin plot lines presented by Lucie-Smith et al. (2020) (blue thin violin lines). In Lucie-Smith et al. (2020) a neural network was trained to directly predict the final halo masses at a voxel-level (using as input the initial density field or the potential). Their network was trained to minimize the difference between predicted and true halo-masses at the particle level, considering only the central particle for the mass predictions in each model evaluation. They train their network in this specific way to answer certain questions regarding halo formation by modifying the input data. We can compare the predictions of our network with Lucie-Smith et al. (2020) taking into account that the way in which we have phrased the problem of halo collapse is completely different. The loss function employed by Lucie-Smith et al. (2020) does not explicitly know about the Lagrangian halo shapes. However, a model that correctly predicts halo masses at the particle level, would need to learn which particle falls onto which halo, explicitly or implicitly. Asking a network to implicitly understand this process – which requires assuming some halo definition – is a task far more difficult than explicitly providing the information of Lagrangian halo shapes during training. In Fig. 8 we can observe that our model generates more precise predictions for the final halo mass at a voxel level since it has been able to explicitly learn the Lagrangian shapes of haloes. This is because it operates under the physical constraint that particles that belong to the same halo have to receive the same halo mass.

In Fig. 9 we present the halo-mass-function (HMF) computed using the validation simulations (solid black line). The dashed black line shows the predicted HMF computed using the results of our instance segmentation pipeline. We can appreciate that our predictions reproduce the N-body results over a range that spans more than two orders of magnitude. Our results improve upon the prediction mass range for the HMF of previous similar approaches (Berger & Stein 2019; Bernardini et al. 2020). This is despite the fact that Bernardini et al. (2020) select their hyper-parameters to reproduce the HMF; while in Berger & Stein (2019) they reproduce the HMF corresponding to `Peak Patch` haloes (Stein et al. 2019), instead of the HMF associated with FoF haloes. In Fig. 9 we also include a solid blue line representing the theoretical HMF predictions using the model by Ondaro-Mallea et al. (2022). We compare this result with the HMF associatted with the haloes predicted by our model using the density and potential fields of a realization with 1024^3 particles and a volume of $V_{\rm box} = (200\,h^{-1}{\rm Mpc})^3$. Both lines show a good agreement in the $10^{12} - 10^{15}\,h^{-1}{\rm M}_\odot$ range.

We conclude that our semantic plus instance segmentation pipeline correctly reproduces the Lagrangian halo shapes of FoF-haloes spanning a mass range between $10^{12}\,h^{-1}{\rm M}_\odot$ and $10^{14.7}\,h^{-1}{\rm M}_\odot$. We have tested the accuracy of our results employing different metrics (presented in several tables and figures). Inferred quantities from our predicted Lagrangian halo regions, such as the predicted halo masses, correctly reproduce the trends computed using N-body simulations and improve upon the results presented in previous studies.

4 EXPERIMENTS

In this section we test how our network reacts to systematic modifications to the input density field and potential and how well it generalizes to scenarios that lie beyond the trained domain. Therefore, we analyze the response to large scale density perturbations, to large scale tidal fields and to changes in the variance of the density field.

4.1 Response to large scale densities

We study the response of the haloes to a large scale over-density such as typically considered in separate universe simulations (Wagner et al. 2015a; Lazeyras et al. 2016; Li et al. 2014). We add a constant δ_ϵ to the input density field $\delta(\vec{q})$ so that the new density field $\delta_*(\vec{q})$ is given by

$$\delta_*(\vec{q}) = \delta(\vec{q}) + \delta_\epsilon, \tag{11}$$

and to maintain consistency with Poisson's equation, see equation (8), we add a quadratic term to the potential:

$$\phi_*(\vec{q}) = \phi(\vec{q}) + \frac{\delta_\epsilon}{6}(\vec{q} - \vec{q}_0)^2 \tag{12}$$

where $\vec{q}_0$ is an arbitrary (and irrelevant) reference point (Stücker et al. 2021b), which we choose to be in the center of our considered domain. Note that we break the periodic boundary conditions here, so that it is difficult to do this operation for the whole box, but instead we consider it only for a smaller region to avoid boundary effects.

We show how haloes respond to this modification in Fig. 10. The middle panel shows the predicted masses associated with the particles/voxels (in a similar way to Fig. 7) for the reference field, $\delta_\epsilon = 0$. The upper and lower panels show the results of including a constant term to the initial over density field of $\delta_\epsilon = -0.5$ and $\delta_\epsilon = 0.5$, respectively.

Increases in the background density lead to more mass collapsing onto haloes, thus generally increasing the Lagrangian volume of haloes. Furthermore, it leads in many cases to previously individual haloes merging to one bigger structure. This is qualitatively consistent with what is observed in separate universe simulations (e.g. Dai et al. 2015; Wagner et al. 2015b; Barreira et al. 2019; Jamieson & Loverde 2019; Terasawa et al. 2022; Artigas et al. 2022).

To evaluate quantitatively whether the model has learned the correct response to large scale density perturbations, we test whether it recovers the same halo bias that has been measured in previous studies (Desjacques et al. 2018, for a review). In separate universe experiments, the linear bias parameter can be inferred as the derivative of the halo mass function with respect to the large-scale density:

$$b_{1L}(M) = \frac{1}{n_h(M)} \frac{\partial n_h(M)}{\partial \delta_\epsilon} \tag{13}$$

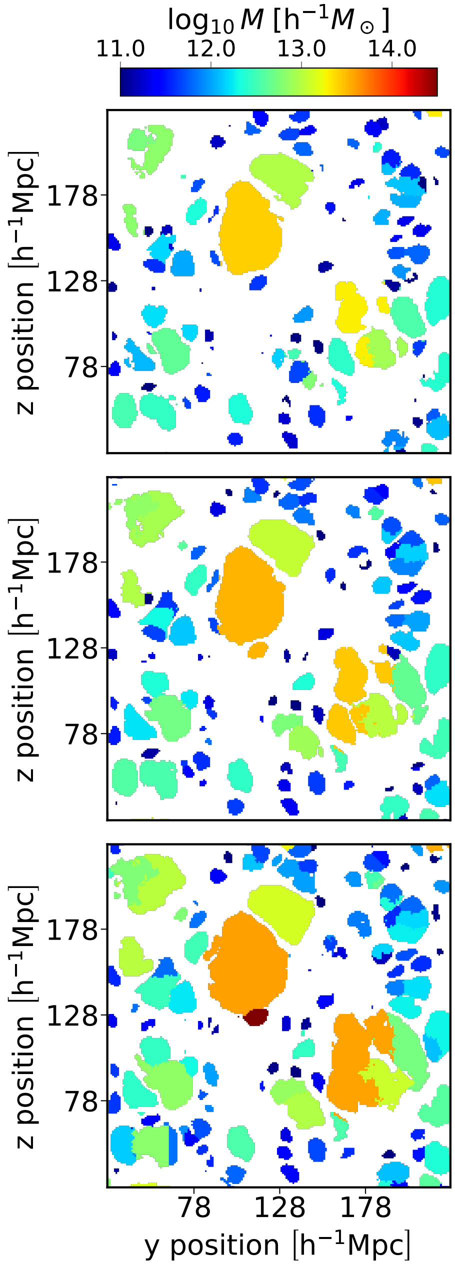

Figure 10. The response of proto-haloes to large scale over-densities. The three panels show over-densities of $\delta_\epsilon = -0.5$, 0 and 0.5 respectively. A larger large scale density tends to increase the Lagrangian volume of haloes and leads to additional mergers in some cases.

Therefore, Lazeyras et al. (2016) used the halo mass function measured in separate universe simulations with different large scale densities δ_ϵ to measure the bias parameters through a finite differences approach. While our qualitative experiment from Figure 11 follows this in spirit, it is difficult to do the same measurement here, since the addition of the quadratic potential term in equation (12) breaks the periodic boundary conditions and makes it difficult to measure the mass function reliably over a large domain. Therefore, we instead adopt an approach to infer the bias from the unperturbed $\delta_\epsilon = 0$ case. Paranjape et al. (2013) shows that the Lagrangian bias parameter can be measured by considering the (smoothed) linear over-density at the

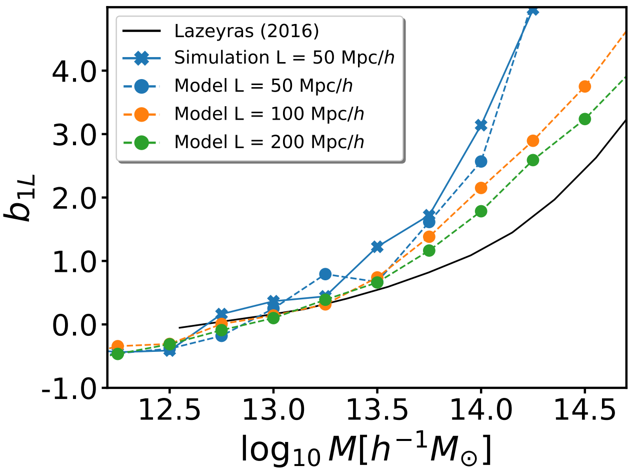

Figure 11. Linear Lagrangian bias parameter b_{1L} for the haloes, measured for different boxsizes L and comparing simulation and model. The model agrees well with the simulation at the $L = 50h^{-1}$Mpc scale, but both are inconsistent with the true large scale bias relation from (Lazeyras et al. 2016) due to effects from the limited size of the simulation volume. Evaluation on larger boxes moves the prediction closer to the known relation, but some deviation is maintained.

Lagrangian location of biased tracers δ_i:

$$b_{1L} = \frac{1}{N}\sum \frac{\delta_i}{\sigma^2} \tag{14}$$

where the sum goes over N different tracers (e.g. all haloes in a given mass bin) and where $\sigma^2 = \langle\delta^2\rangle$ is the variance of the (smoothed) linear density field. Since this measurement should give meaningful results only on reasonably large scales, we smooth the Lagrangian density field with a Gaussian kernel with width $\sigma_r = 6h^{-1}$Mpc. We measure the smoothed linear density δ_i at the Lagrangian center of mass of each halo patch and then we measure the bias by evaluating equation (14) in different mass bins.

We show the resulting b_{1L} as a function of mass in Figure 11. The blue solid and dashed lines show the bias parameters measured in an $L = 50h^{-1}$Mpc box for the simulated versus predicted halo patches respectively. These two seem consistent, showing that the model has correctly learned the bias relation that is captured inside of the training set. However, this ($L = 50h^{-1}$Mpc) relation is not consistent with the well-measured relation from larger scale simulations, indicated as black solid line adopted from Lazeyras et al. (2016). This is because very massive haloes $M \gg 10^{14}h^{-1}M_\odot$ do not form in simulations of such a small volume, but they are important to get the correct bias of smaller mass haloes, since wherever a large halo forms, no smaller halo can form. Our network has never seen such large scales, so it is questionable whether it has any chance of capturing the large scale bias correctly. However, it might be that what it has learned in the small scale simulation transfers to larger scales. To test this, we evaluate the network on two larger boxes, $L = 100h^{-1}$Mpc and $L = 200h^{-1}$Mpc, shown as orange and green lines in Figure 11. Clearly, these cases match the true bias relation better, but still show some significant deviation e.g. at $M \sim 10^{14}h^{-1}M_\odot$. Therefore, we conclude that the network generalizes only moderately well to larger scales and halo masses. Improved performance could possibly be achieved by extending the training set to larger simulations and by increasing the field of view of the network.

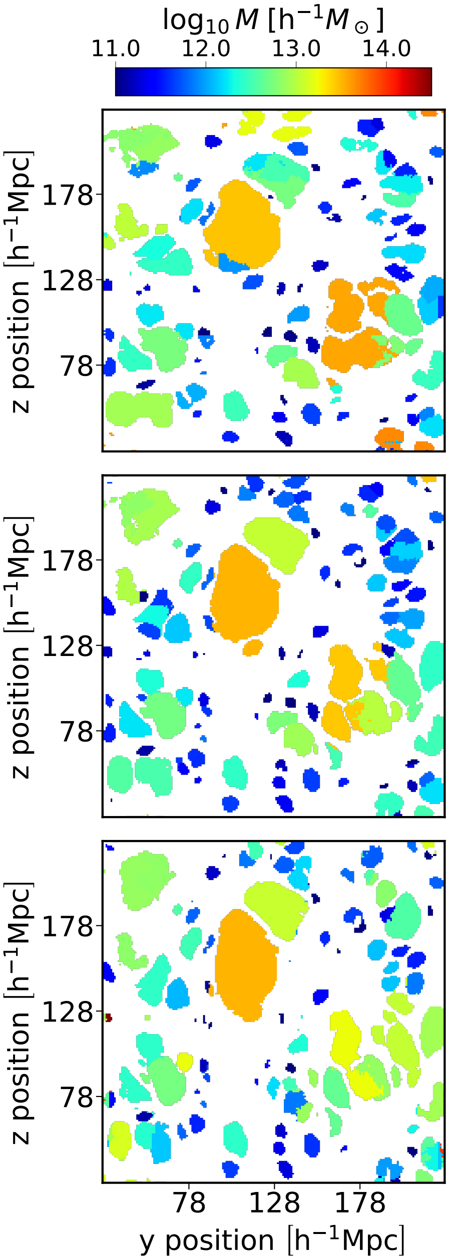

Figure 12. The response of proto-halo regions towards a large scale tidal field. The different panels show the cases with $\lambda_z = -0.5$, 0 and 0.5 – corresponding to a stretching tidal field, no tidal field and a compressing tidal field in the vertical direction respectively. A negative (stretching) tidal field delays infall and shrinks the proto-halo patches in the corresponding direction, whereas a positive (compressing) tidal field facilitates infall and extends the proto-halo patches.

4.2 Response to large scale tidal fields

In a second experiment we want to study the response of haloes to purely anisotropic changes of the initial conditions, by adding a large scale tidal field. We therefore, aim to emulate a modification similar to the ones considered in anisotropic separate universe simulations (Schmidt et al. 2018; Stücker et al. 2021a; Masaki et al. 2020; Akitsu et al. 2021). We modify the input potential through the term

$$\phi_*(\vec{q}) = \phi(\vec{q}) + \frac{1}{2}(\vec{q} - \vec{q}_0)^T T(\vec{q} - \vec{q}_0) \tag{15}$$

$$T = \begin{pmatrix} 0 & 0 & 0 \\ 0 & -\lambda_z & 0 \\ 0 & 0 & \lambda_z \end{pmatrix} \tag{16}$$

Since we are considering a trace-free tidal tensor, we do not need to include any modifications to the initial density field. The results of introducing the tidal field are presented Fig. 12. In the upper panel in which we have imposed a value of $\lambda_z = -0.5$, the regions of typical proto-haloes are slightly reduced in the z-direction and extended in the y-direction. Further, in some cases haloes merge additionally in the y-direction while separating in the z-direction. In the bottom panel with $\lambda_z = 0.5$ we observe the opposite behavior, with proto-halo shapes elongated in the z-direction and reduced in the y-direction. These observations are consistent with the naive expectation: A positive λ_z means a contracting tidal field in the z-direction, which facilitates infall in this direction, whereas a negative λ_z delays the infall. Therefore, proto-haloes appear extended in the direction where the tidal field has a contracting effect. This should not be confused with the response of the halo shapes in Eulerian space which has the opposite behavior – reducing the halo's extend in the direction where the tidal field is contracting (Stücker et al. 2021a). Therefore, a large scale tidal field effects that *the direction from which more material falls in, is the direction where the final halo is less extended*.

However, by comparing Figures 10 and 12, we note that the effect of modifying the eigenvalues of the tidal tensor (while keeping the trace fixed) is much less significant than modifying its trace δ by a similar amount. Modifying δ leads to strong differences in the abundance and the masses of haloes whereas the modifications to the tidal field strongly affects the shapes, but has a much smaller effect on typical masses – if at all.

We note that our results here seem partially at odds with the observations made by Lucie-Smith et al. (2020). In that study, a convolutional neural network is trained to predict from the initial conditions for each particle the mass of the halo which it will become part of. The authors find that anisotropic features of the initial conditions do not enhance this prediction beyond what is possible to predict already from pure spherical averages. Optimally predicting a particle's halo mass implicitly requires assigning it to the correct halo, so that we do not quite understand why anisotropic features appear irrelevant in their work, while they appear relevant in our work here. However, our method has an improved performance on predicting halo masses and seems to extract the available information more reliably. We suggest that our approach benefits from imposing the physical constraint that different particles of the same halo must be assigned the same mass. Such a constraint cannot be enforced in the standard extended Press-Schechter approach (Bond et al. 1991) so that the conclusion of Lucie-Smith et al. (2020) may still apply to what is possible within the excursion set framework. However, more generally, we conclude that *anisotropic features of the initial conditions are clearly relevant to decide which particles become part of which haloes*.

Finally, we note that the response of the Lagrangian haloes shape is particular interesting in the context of tidal torque theory (White 1984). To predict the angular momentum of haloes, tidal torque theory requires knowledge of both the tidal tensor and the Lagrangian inertia tensor of haloes. Further, it has been argued that the misalignment of tidal field and Lagrangian inertia tensor is a key factor for predicting galaxy properties (Moon & Lee 2023). Our experiments show that modifications of the tidal tensor itself also trigger modifications of the Lagrangian shape. Precisely understanding this relation would be relevant to correctly predict halo spins from the initial conditions. Note that such responses are inherently absent in most density based structure formation models (e.g. Press & Schechter 1974; Bond et al. 1991; Sheth & Tormen 2002), but could possibly be accounted for by recently proposed approaches based on the Lagrangian potential (Musso & Sheth 2021a, 2023a).

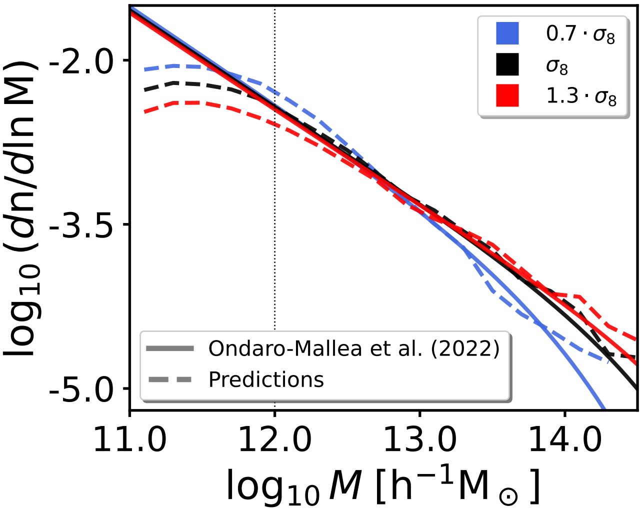

Figure 13. Comparison of HMF predictions with variations in the cosmological parameter σ_8. Solid lines represent HMF predictions from Ondaro-Mallea et al. (2022). Dashed lines indicate our model's predictions. Blue and red curves correspond to scenarios with $\sigma_8 = 0.5802$ and $\sigma_8 = 1.077$ respectively. Black lines show the results for $\sigma_8 = 0.8288$ (our reference cosmology).

4.3 Response to changes in the variance of the density field

We now study whether our model can generalize to scenarios different from the training set by investigating how it responds to variations in σ_8, deviating 30% from the original Planck Collaboration et al. (2020) cosmology. Our aim is to discern if the network, trained on a singular variance setting, has gained enough insight into halo formation to anticipate outcomes considering different values for the variance of the initial density field. These modifications only affect the initial conditions which are fully visible to the network, so that it could be possible that the network correctly extrapolates to these scenarios.

In Fig. 13 we show how the HMF reacts to changes in σ_8 in comparison to the measured mass functions from Ondaro-Mallea et al. (2022) (solid lines) as a benchmark. Our predictions for the HMF (dashed lines) are generated by taking the average results of 10 different boxes, each one spanning $L = 50h^{-1}$Mpc, with σ_8 values set to 0.5802 (blue lines), 0.8288 (black lines), and 1.077 (red lines). The model's predictions reveal a discrepancy with the anticipated HMF behavior beneath the threshold of $\sim 10^{12.7}h^{-1}M_\odot$ for both $\sigma_8 \approx 0.5802$, and $\sigma_8 \approx 1.077$. This discrepancy is attributed to the model's training on datasets characterized by the specific σ_8 from Planck Collaboration et al. (2020). The model's ability to extrapolate to different variances remains limited. At higher masses, however, the network's predictions correspond more closely with the expected HMF. This partial alignment suggests that the network possesses some degree of generalization capability. Nonetheless, for reliable application across varying cosmologies, incorporating these scenarios into the training set is essential.

5 DISCUSSION & CONCLUSIONS

We present a novel approach to understand and predict halo formation from the initial conditions employed in N-body simulations. Benchmark tests indicate that our model can predict Lagrangian FoF-halo regions for simulations efficiently, taking around 7 minutes in a GPU for a simulation with 256^3 particles in a volume of $50h^{-1}$Mpc. For those interested in leveraging or further enhancing our work, we have make our codes publicly available: https://github.com/daniellopezcano/instance_halos.

Our model consist of a semantic network that reliably recognizes regions in Lagrangian space where haloes form, and an instance segmentation network, that identifies individual haloes from the semantic output. Our predictions accurately reproduce simulation results and outperform traditional analytical, semi-analytical techniques, and prior ML methods.

The foundation for our instance segmentation model is the Weinberger approach, first introduced by De Brabandere et al. (2017). This technique lets us develop a more general framework for identifying Lagrangian halo patches than previous attempts. Employing the Weinberger loss approach, we bypass some limitations of other instance segmentation methods, like the watershed technique employed by Bernardini et al. (2020). With our approach we manage to predict the complicated Lagrangian shapes of haloes that are formed in N-body simulations. This is notably more difficult than the predictions of spherical Peak-Patch-haloes that were considered by Berger & Stein (2019).

Additionally, we quantify in how far halo formation is indetermined by the resolved scales of the initial conditions, to establish an optimal performance limit of machine learning methods. We infer this limit by comparing two simulations which only differ in their initial conditions realization on scales beyond the resolution level. We find an agreement between our model predictions and reference simulations similar to the agreement between the two 'baseline' simulations. This shows that our model extracts information encoded in the initial conditions close to optimal. We suggest that such reference experiments may also be used as a baseline in other ML studies to establish whether information is extracted optimally.

Upon evaluating our semantic model, we measure an accuracy of 0.864 and an F_1-score of 0.838. Compared to the baseline simulations, which have an accuracy of 0.865 and an F_1-score of 0.848, our model results stand remarkably close, demonstrating its capability to predict halo regions nearly matching N-body simulations' natural variability.

We also assess our instance segmentation network using various metrics. As depicted in Fig. 8, our model closely aligns with the baseline across a broad mass range, outperforming previous methods like Lucie-Smith et al. (2020) which did not exploit the physical constraints of the problem at hand[3]. Moreover, the halo mass function (HMF) predictions in Fig. 9 closely match the true ground truth values across three orders of magnitude. The visual representations in Fig. 7 reinforce our model's precision, faithfully replicating Lagrangian halo patch positions and shapes.

We have tested through experiments how the network reacts to systematic modifications of the initial conditions. We find that the network correctly captures the response to density perturbations at the finite boxsize provided in the training set. However, it struggles to generalize to larger boxsizes and to cosmologies with different amplitudes of the density field σ_8. This can easily be improved by increasing the diversity of the training set.

Further, we have found that our network utilizes information from the potential field that is not encoded in the density field of any finite region. Modifications to a large scale tidal field are consistent with the same linear density field, but do affect the potential landscape. Our network predicts that such tidal fields affect the Lagrangian shape of haloes in an anisotropic manner which is consistent with the intuitive expectation of how a tidal field accelerates and decelerates the infall anisotropically. This is in contrast to the study by Lucie-Smith et al. (2020) in which a (differently trained) network appeared to be insensitive to anisotropic aspects of the initial conditions.

We have demonstrated the robustness of our model in its current applications and we believe it could find potential utility in several other scenarios like: crafting emulated merger trees, aiding separate-universe style experiments (e.g. Lazeyras et al. 2016; Stücker et al. 2021a) and informing the development of analytical methods for halo formation (e.g. Musso & Sheth 2021b, 2023b). Further, it may be used to help understanding the development of spin and intrinsic alignments in haloes and galaxies by establishing how tidal fields modify the Lagrangian shapes of haloes. This is a vital ingredient to predict the spin of haloes through tidal torque theory (White 1984). We encourage experts in these fields to use our open-source code as a basis for tackling and exploring these and other related problems.

The findings presented in this work are promising but there exist some aspects of our models that would benefit from further investigation. For instance, extending our methodology to understand other halo properties beyond mass would be a logical next step. It would also be interesting to test our model's performance under a wider variety of simulation conditions, including variations in cosmology and redshift. An additional avenue of exploration might involve delving into capturing intricate structural details, specifically the gap features in the predicted Lagrangian halo regions. Generative Adversarial Networks (GANs) are tools that have demonstrated potential in reproducing data patterns in the context of cosmological simulations (e.g. Rodríguez et al. 2018; Villaescusa-Navarro et al. 2021; Schaurecker et al. 2021; Robles et al. 2022; Nguyen et al. 2023; Zhang et al. 2023). Hence, employing a GAN-like approach might help recreating these gap features, further improving our model's ability to mimic the structures of haloes found in N-body simulations.

In conclusion, this study showcases the potential of machine learning for facilitating the study of halo formation processes in the context of cosmological N-body simulations. We provide a fast model that exploits the available information close to optimally. We hope our approach serves as a useful tool for researchers working with N-body simulations, opening avenues for future advancements.

ACKNOWLEDGEMENTS

DLC, JS, MPI and REA acknowledge the support of the ERC-StG number 716151 (BACCO) and of project PID2021-128338NB-I00 from the Spanish Ministry of Science. DLC further thanks Lurdes Ondaro-Mallea for helping to run the baseline simulations and Luisa Lucie-Smith, Drew Jamieson, Francisco Maion, Matteo Zennaro, and Daniel Muñoz-Segovia for their helpful discussions and comments. This work used the following software: Python,

[3] In Lucie-Smith et al. (2020) different particles that belong to the same halo could be assigned different mass predictions.

Matplotlib (Hunter 2007), TensorFlow (Abadi et al. 2015), and Numpy (van der Walt et al. 2011).

DATA AVAILABILITY

The data underlying this article will be shared on reasonable request to the corresponding author.

REFERENCES

Abadi M., et al., 2015, TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, `https://www.tensorflow.org/`

Akitsu K., Li Y., Okumura T., 2021, J. Cosmology Astropart. Phys., 2021, 041

Alves de Oliveira R., Li Y., Villaescusa-Navarro F., Ho S., Spergel D. N., 2020, arXiv e-prints, p. arXiv:2012.00240

Andres-San Roman J. A., et al., 2023, bioRxiv, pp 2023–01

Angulo R. E., Hahn O., 2022, Living Reviews in Computational Astrophysics, 8, 1

Angulo R. E., Springel V., White S. D. M., Jenkins A., Baugh C. M., Frenk C. S., 2012, Monthly Notices of the Royal Astronomical Society, 426, 2046

Angulo R. E., Zennaro M., Contreras S., Aricò G., Pellejero-Ibañez M., Stücker J., 2021, Monthly Notices of the Royal Astronomical Society, 507, 5869

Arnab A., Torr P. H. S., 2017, arXiv e-prints, p. arXiv:1704.02386

Artigas D., Grain J., Vennin V., 2022, J. Cosmology Astropart. Phys., 2022, 001

Bai M., Urtasun R., 2016, arXiv e-prints, p. arXiv:1611.08303

Barreira A., Nelson D., Pillepich A., Springel V., Schmidt F., Pakmor R., Hernquist L., Vogelsberger M., 2019, MNRAS, 488, 2079

Berger P., Stein G., 2019, MNRAS, 482, 2861

Bernardini M., Mayer L., Reed D., Feldmann R., 2020, MNRAS, 496, 5116

Betts J. C., van de Bruck C., Arnold C., Li B., 2023, arXiv e-prints, p. arXiv:2305.02122

Bond J. R., Cole S., Efstathiou G., Kaiser N., 1991, ApJ, 379, 440

Chacón J., Vázquez J. A., Almaraz E., 2022, Astronomy and Computing, 38, 100527

Dai L., Pajer E., Schmidt F., 2015, J. Cosmology Astropart. Phys., 2015, 059

Davis M., Efstathiou G., Frenk C. S., White S. D. M., 1985, ApJ, 292, 371

De Brabandere B., Neven D., Van Gool L., 2017, arXiv e-prints, p. arXiv:1708.02551

Deng R., Shen C., Liu S., Wang H., Liu X., 2018, arXiv e-prints, p. arXiv:1807.10097

Desjacques V., Jeong D., Schmidt F., 2018, Phys. Rep., 733, 1

Eisenstein D. J., Hut P., 1998, ApJ, 498, 137

Franco-Barranco D., Muñoz-Barrutia A., Arganda-Carreras I., 2021, Neuroinformatics

Franco-Barranco D., Andrés-San Román J. A., Gómez-Gálvez P., Escudero L. M., Muñoz-Barrutia A., Arganda-Carreras I., 2023, in 2023 IEEE 20th International Symposium on Biomedical Imaging (ISBI). pp 1–5

Frenk C. S., White S. D. M., 2012, Annalen der Physik, 524, 507

Fukushima K., 1980, Biological Cybernetics, 36, 193

Genel S., et al., 2019, ApJ, 871, 21

Giusarma E., Reyes Hurtado M., Villaescusa-Navarro F., He S., Ho S., Hahn C., 2019, arXiv e-prints, p. arXiv:1910.04255

Gunn J. E., 1977, ApJ, 218, 592

Gunn J. E., Gott J. Richard I., 1972, ApJ, 176, 1

Hagberg A., Swart P., S Chult D., 2008, Technical report, Exploring network structure, dynamics, and function using NetworkX. Los Alamos National Lab.(LANL), Los Alamos, NM (United States)

Hahn O., Abel T., 2011, MNRAS, 415, 2101

He K., Zhang X., Ren S., Sun J., 2015, arXiv e-prints, p. arXiv:1512.03385

He S., Li Y., Feng Y., Ho S., Ravanbakhsh S., Chen W., Póczos B., 2019, Proceedings of the National Academy of Science, 116, 13825

Hunter J. D., 2007, Computing in Science & Engineering, 9, 90

Jamieson D., Loverde M., 2019, Phys. Rev. D, 100, 123528

Jamieson D., Li Y., He S., Villaescusa-Navarro F., Ho S., Alves de Oliveira R., Spergel D. N., 2022, arXiv e-prints, p. arXiv:2206.04573

Jiang F., van den Bosch F. C., 2014, MNRAS, 440, 193

Kirillov A., Levinkov E., Andres B., Savchynskyy B., Rother C., 2016, arXiv e-prints, p. arXiv:1611.08272

Kirillov A., He K., Girshick R., Rother C., Dollár P., 2018, arXiv e-prints, p. arXiv:1801.00868

Kirillov A., et al., 2023, arXiv e-prints, p. arXiv:2304.02643

Krizhevsky A., Sutskever I., Hinton G. E., 2012, in Pereira F., Burges C., Bottou L., Weinberger K., eds, Vol. 25, Advances in Neural Information Processing Systems. Curran Associates, Inc., `https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf`

Lacey C., Cole S., 1993, MNRAS, 262, 627

Lazeyras T., Wagner C., Baldauf T., Schmidt F., 2016, J. Cosmology Astropart. Phys., 2016, 018

Lecun Y., Bottou L., Bengio Y., Haffner P., 1998, Proceedings of the IEEE, 86, 2278

Li Y., Hu W., Takada M., 2014, Phys. Rev. D, 89, 083519

Lin Z., et al., 2021, in Medical Image Computing and Computer Assisted Intervention–MICCAI 2021: 24th International Conference, Strasbourg, France, September 27–October 1, 2021, Proceedings, Part I 24. pp 164–174

Long J., Shelhamer E., Darrell T., 2014, arXiv e-prints, p. arXiv:1411.4038

Lucie-Smith L., Peiris H. V., Pontzen A., Lochner M., 2018, MNRAS, 479, 3405

Lucie-Smith L., Peiris H. V., Pontzen A., 2019, MNRAS, 490, 331

Lucie-Smith L., Peiris H. V., Pontzen A., Nord B., Thiyagalingam J., 2020, arXiv e-prints, p. arXiv:2011.10577

Lucie-Smith L., Barreira A., Schmidt F., 2023, MNRAS, 524, 1746

Ludlow A. D., Porciani C., 2011, MNRAS, 413, 1961

Masaki S., Nishimichi T., Takada M., 2020, MNRAS, 496, 483

Meyer F., 1994, Signal Processing, 38, 113

Moon J.-S., Lee J., 2023, arXiv e-prints, p. arXiv:2311.03632

Musso M., Sheth R. K., 2021a, MNRAS, 508, 3634

Musso M., Sheth R. K., 2021b, MNRAS, 508, 3634

Musso M., Sheth R. K., 2023a, MNRAS, 523, L4

Musso M., Sheth R. K., 2023b, MNRAS, 523, L4

Nguyen T., Modi C., Yung L. Y. A., Somerville R. S., 2023, arXiv e-prints, p. arXiv:2308.05145

Ondaro-Mallea L., Angulo R. E., Zennaro M., Contreras S., Aricò G., 2022, MNRAS, 509, 6077

Otsu N., 1979, IEEE transactions on systems, man, and cybernetics, 9, 62

Paranjape A., Sefusatti E., Chan K. C., Desjacques V., Monaco P., Sheth R. K., 2013, MNRAS, 436, 449

Peebles P. J. E., 1980, The large-scale structure of the universe

Perraudin N., Srivastava A., Lucchi A., Kacprzak T., Hofmann T., Réfrégier A., 2019, Computational Astrophysics and Cosmology, 6, 5

Planck Collaboration et al., 2020, A&A, 641, A6

Press W. H., Schechter P., 1974, ApJ, 187, 425

Robles S., Gómez J. S., Ramírez Rivera A., Padilla N. D., Dujovne D., 2022, MNRAS, 514, 3692

Rodríguez A. C., Kacprzak T., Lucchi A., Amara A., Sgier R., Fluri J., Hofmann T., Réfrégier A., 2018, Computational Astrophysics and Cosmology, 5, 4

Ronneberger O., Fischer P., Brox T., 2015, arXiv e-prints, p. arXiv:1505.04597

Schanz A., List F., Hahn O., 2023, arXiv e-prints, p. arXiv:2310.06929

Schaurecker D., Li Y., Tinker J., Ho S., Refregier A., 2021, arXiv e-prints, p. arXiv:2111.06393

Schmidt A. S., White S. D. M., Schmidt F., Stücker J., 2018, MNRAS, 479, 162

Sheth R. K., Tormen G., 2002, MNRAS, 329, 61

Sheth R. K., Mo H. J., Tormen G., 2001, MNRAS, 323, 1

Sousbie T., 2011, MNRAS, 414, 350

Springel V., 2015, N-GenIC: Cosmological structure initial conditions (ascl:1502.003)

Springel V., White S. D. M., Tormen G., Kauffmann G., 2001, MNRAS, 328, 726

Springel V., et al., 2008, MNRAS, 391, 1685

Stein G., Alvarez M. A., Bond J. R., 2019, MNRAS, 483, 2236

Stücker J., Schmidt A. S., White S. D. M., Schmidt F., Hahn O., 2021a, MNRAS, 503, 1473

Stücker J., Angulo R. E., Busch P., 2021b, MNRAS, 508, 5196

Szegedy C., et al., 2014, arXiv e-prints, p. arXiv:1409.4842

Terasawa R., Takahashi R., Nishimichi T., Takada M., 2022, Phys. Rev. D, 106, 083504

Tierny J., Favelier G., Levine J. A., Gueunet C., Michaux M., 2017, IEEE Transactions on Visualization and Computer Graphics (Proc. of IEEE VIS)

Villaescusa-Navarro F., et al., 2021, ApJ, 915, 71

Virtanen P., et al., 2020, Nature Methods, 17, 261

Wagner C., Schmidt F., Chiang C. T., Komatsu E., 2015a, MNRAS, 448, L11

Wagner C., Schmidt F., Chiang C. T., Komatsu E., 2015b, MNRAS, 448, L11

Wei D., et al., 2020, in International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI). pp 66–76

Weinberger K. Q., Saul L., 2009, J. Mach. Learn. Res., 10, 207–244

White S. D. M., 1984, ApJ, 286, 38

Wu Z., et al., 2021, ApJ, 913, 2

Xie S., Tu Z., 2015, arXiv e-prints, p. arXiv:1504.06375

Zhang X., Lachance P., Ni Y., Li Y., Croft R. A. C., Di Matteo T., Bird S., Feng Y., 2023, arXiv e-prints, p. arXiv:2305.12222

van der Walt S., Colbert S. C., Varoquaux G., 2011, Computing in Science Engineering, 13, 22

APPENDIX A: WATERSHED SEGMENTATION

In this appendix we present an alternative approach to instance segmentation, based on the watershed approach. Originally we have tried this technique to address the instance segmentation problem, but we finally decided to use the Weinberger approach presented in the main paper because of its theoretical advantages. These are that the loss function closer reflects the objective, that it is possible to predict disconnected regions, and that it is not necessary to define borders. However, during our exploration we have gained some insights of how to make watershed based instance segmentation techniques work for friends-of-friends proto-haloes. We will explain these here for the benefit of future studies.

Our watershed approach make use of a U-Net-based architecture Ronneberger et al. (2015), specifically a 3D Residual U-Net based on previous work Franco-Barranco et al. (2021). The model's input consisting of $128 \times 128 \times 128 \times 2$ voxels for $(x, y, z, channels)$ axes. The two input *channels* correspond to initial density field and the potential.

The model is trained to predict two output channels: binary foreground segmentation masks and instance contours masks. Following the prediction, the two outputs are thresholded (automatically using Otsu's method Otsu (1979)) and combined. Next, a connected components operation is applied to generate distinct, non-touching halo instance seeds. Subsequently, a marker-controlled watershed algorithm Meyer (1994) is applied, using three key components: 1) the inverted foreground probabilities as the input image (representing the topography to be flooded), 2) the generated instance seeds as the marker image (defining starting points for the flooding process), and 3) a binarized version of the foreground probabilities as the mask image (constraining the extent of object expansion). To binarize the latter, we employed a threshold value of 0.372, which was determined through the application of the identical methodology outlined in Appendix C. The collective implementation of these components facilitates the creation of individual halo instances (see Fig. A1 for a visual representation). This strategy has been extensively employed within the medical field with remarkable success Wei et al. (2020); Lin et al. (2021); Andres-San Roman et al. (2023).

In order to facilitate the generation of the two channels used to train the network, several transformations were applied to the labels. For each halo instance, small particles along the edges were removed, central holes were filled, and the labels were dilated by one pixel. This process results in instances with smoother boundaries, thereby aiding the network in training (see Fig. A2).

The result of this method is depicted in Fig. A3. The code is open source and readily available in BiaPy Franco-Barranco et al. (2023).

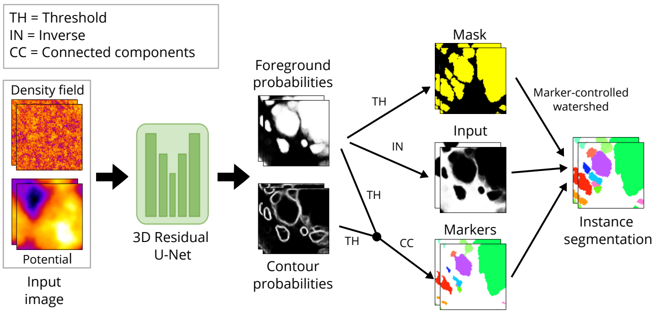

Figure A1. Processing pipelines of our watershed segmentation approach. The input 3D image contains two channels: the density field and the potential. The model predicts foreground and contour probabilities that are fused to create three inputs for a marker-controlled watershed to produce individual instances.

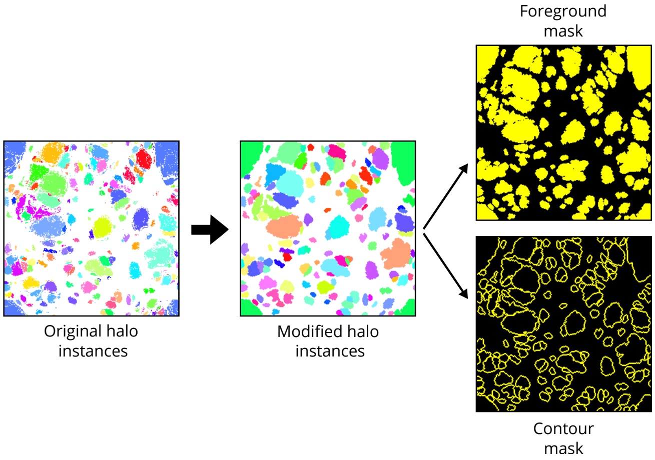

Figure A2. Data preparation process of our watershed segmentation approach. From left to right: the original halo instances for the considered prediction problem, subsequent modifications involving the removal of small holes and spurious pixels and contour smoothing, and the presentation of both the foreground and contour masks utilized for model training. Pixels coloured in white do not belong to any halo. Pixels with the same colour belong to the same halo and different colours indicate different haloes.

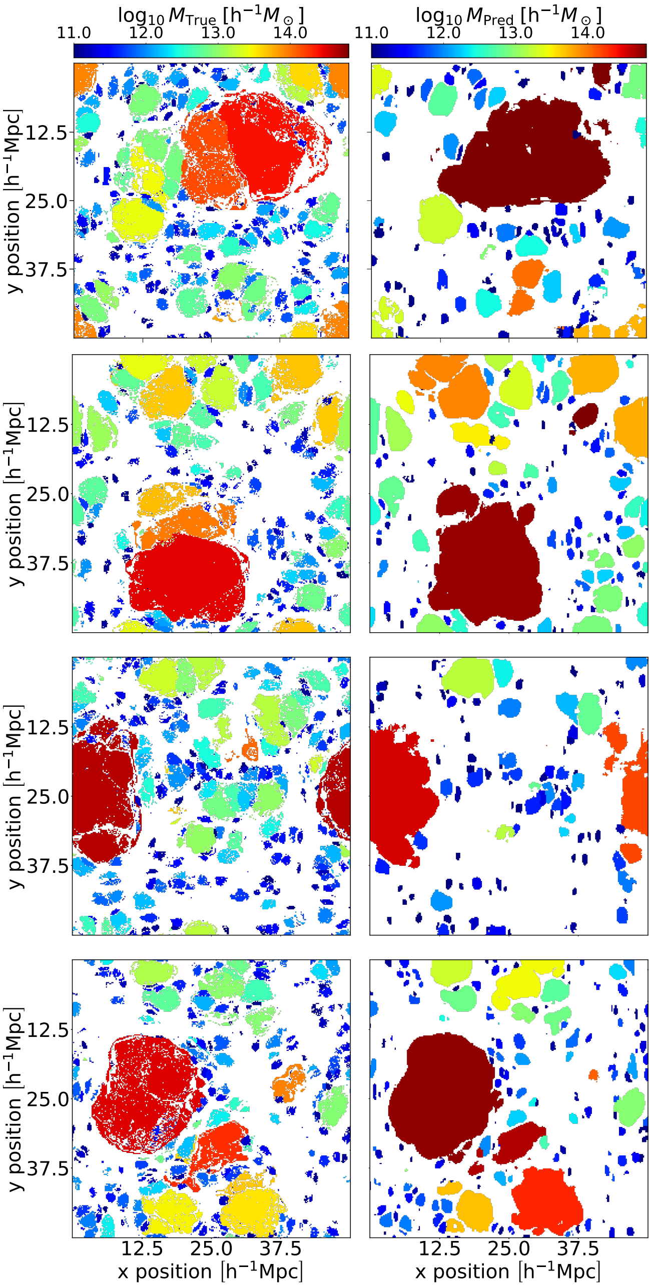

Figure A3. Results of our watershed segmentation approach presented in an analogous way to results from Fig. 7.

APPENDIX B: CLUSTERING ALGORITHM

In this appendix, we describe the clustering algorithm that we have developed. This algorithm calculates instance predictions from the pseudo-space representations that are output by our instance segmentation network.

As described in §§2.2, the output of our instance network consist of a set of points that populate an abstract space (referred to as pseudo-space). Our instance network has been trained to minimize the Weinberger loss function 7, hence, we expect that the predicted mapping of points in the pseudo-space causes that points corresponding to the same instances to be close to each other, and separated to points that correspond to different instances. In the ideal case where $\mathcal{L}_{\rm Wein}$, all points belonging to the same instance would be no farther apart from each other than a distance $2 \cdot \delta_{\rm Pull}$, and the points corresponding to separate instances would be, as close as a distance $2 \cdot \delta_{\rm Push} - \delta_{\rm Pull}$ close to each other. However, we cannot expect that our network always separates perfectly the different instances. For example, if some Lagrangian voxel has a 60% chance to belong to halo A and a 40% chance to belong to halo B, then the optimal location in pseudo space (that statistically minimizes the loss) may be somewhere in between the center of halo A and B in pseudo space and not inside the $\delta_{\rm Pull}$ radius of neither. Therefore, we employ a clustering algorithm that can segment the pseudo-space distribution of points also when $\mathcal{L}_{\rm Wein}$ is not exactly zero.

For this we first estimate the local pseudo-space density ρ_i for each point i. For this we compute the distance $r_{k,i}$ to the kth-nearest neighbour of the point and assign

$$\rho_i = \frac{3k}{4\pi r_{k,i}^3} \tag{B1}$$

where $k = N_{\rm dens}$ is a hyper-parameter of the clustering algorithm. We accelerate this step with the CKD-TREE from the SCIPY package in PYTHON (Virtanen et al. 2020).

Then we determine groups as the descending manifold of the maxima that exceed a persistence ratio threshold $\rho_{\rm max}/\rho_{\rm sad} \geq p_{\rm thresh}$ between maximum and saddle-point. The descending manifold corresponds to the set of particles from whose location following the local density gradient would end up in the same maximum (e.g. Sousbie 2011; Tierny et al. 2017). For this we use a slightly modified version of the density segmentation algorithm used in SUBFIND (Springel et al. 2001):

We consider the particles from highest to lowest density. For each particle we consider from the $N_{\rm ngb}$ nearest particles the subset of particles that have a higher density than ρ_i (this set may be empty). Among these we select the set B_i of the (up to) two closest particles. This set can have zero, one or two particles.

- If the set B_i is empty, then there is a density maximum $\rho_{\rm max} = \rho_i$ and we start growing a new subgroup around it.
- If the set B_i contains a single particle, or two particles that are of the same group, the particle i is attached to the corresponding group.
- If B_i contains two particles of different groups, then i is potentially a saddle-point. We check whether the group with the lower density maximum $\rho_{\rm max}$ has a sufficient persistence $\rho_{\rm max}/\rho_i \leq p_{\rm thresh}$. If not, then we merge the two groups (and keep the denser maximum). Otherwise we keep both groups and we assign the particle to the group of the denser particle in B_i. (This step corresponds to following the local discrete density gradient.)

Note that unlike the SUBFIND algorithm, we merge groups not at every saddle-point, but only if they are below a persistence threshold. Therefore, sufficiently persistent groups are grown beyond their saddle-point and ultimately correspond to the descending manifold of their maximum.

The clustering algorithm has three hyper-parameters $N_{\rm dens}$, $N_{\rm ngb}$ and $p_{\rm thresh}$. We have done a hyper-parameter optimization over these and found that $N_{\rm dens} = 20$, $N_{\rm ngb} = 15$ (quite close to the default parameters in the SUBFIND algorithm, 20 and 10 respectively) and $p_{\rm thresh} = 4.2$ give the best results, though our results are not very sensitive to moderate deviations from this. We can understand the quantitative value of the persistence ratio threshold by considering that the relative variance of our density estimate is

$$\sigma_{\log \rho} \approx \frac{\sigma_\rho}{\rho} = \frac{1}{\sqrt{N_{\rm dens}}} \approx 0.22 \tag{B2}$$

so that at a fixed background density having a density contrast of $p_{\rm thresh} = 4.2$ due to Poisson noise corresponds to a

$$\Delta \log \rho = \log(p_{\rm thresh}) \approx 1.43 \approx 6.5\sigma_{\log \rho} \tag{B3}$$

outlier. Therefore, the persistence ratio threshold $p_{\rm thresh}$ ensures that it is very unlikely that our algorithm mistakes a spurious overdensity in the pseudo space for a group.

APPENDIX C: SEMANTIC THRESHOLD

In the bottom panel of Fig. C1 we present how the predicted fraction of voxels that are members of a halo (that is $1-\beta$) evolves as we change the semantic threshold (black solid line). As it can be expected, when the semantic threshold is close to zero, the majority of voxels are identified as members of haloes, and the contrary occurs when the semantic threshold approximates one. The horizontal dashed-dotted line corresponds to the ground truth value of $1-\beta = 0.418$, measured in the validation simulations. The semantic threshold value that we have selected is 0.589 (black dotted vertical line). This value corresponds to the intersection between the black solid line and the dashed-dotted line; it ensures that the total fraction of voxels that are members of haloes is correctly reproduced. Choosing this criterion to determine the semantic threshold also ensures more robust instance predictions since the number of FP cases is reduced, hence eliminating potentially uncertain pseudo-space particles that would complicate the clustering procedure.

In the top panel of Fig. C1 we show the evolution of several metrics as a function of the semantic threshold value. These metrics allow us to asses the quality of our semantic predictions by comparing our results with values obtained using the baseline simulations. We study the behaviour of five different metrics: True Positive Rate TPR, True Negative Rate TNR, Positive Predictive Value PPV, Accuracy ACC and the F_1-score.

In the top panel of Fig. C1 we also present the values obtained for the different metrics using the baseline simulations (horizontal dashed lines). We have obtained these results considering one of the baseline simulations as predicted maps and the other simulation as the ground truth. The values measured for the different metrics in the baseline simulations give us an expected ideal performance that we would like to reproduce with our model.

If we focus on the performance curves for the accuracy and the F_1-score (orange and yellow lines respectively) we can appreciate that they always remain under the baseline limit. The curve for the F_1-score peaks around the value for the semantic threshold of 0.5, which is a behaviour we expected since we considered the balanced cross-entropy loss to train our semantic model. The value for the F_1-score at its maximum is $\mathrm{F}_1(0.5) = 0.842$, which is very similar to the value at the point in which we have fixed the semantic threshold, $\mathrm{F}_1(0.589) = 0.838$. The F_1-score obtained is only about 5% away from the optimal value obtained from the baseline simulations $\mathrm{F}_1^{\rm Chaos} = 0.884$. The accuracy reaches its maximum value around the semantic threshold of 0.58, where $\mathrm{ACC}(0.58) = 0.864$; the value for the model accuracy is even closer to the baseline limit $\mathrm{ACC}^{\rm Chaos} = 0.903$.

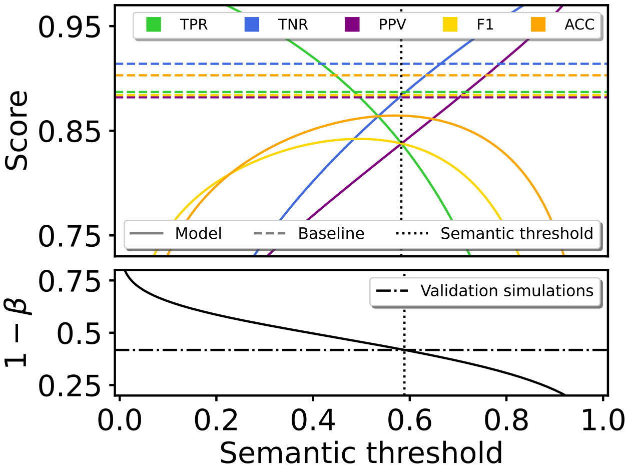

Figure C1. Top panel: Evolution of different metrics (TPR - green, TNR - blue, PPV - purple, F_1-score - yellow & ACC - orange) measured employing the predictions of the semantic model as a function of the semantic threshold selected (solid lines); we also show the values measured for the corresponding metrics studying the differences between the baseline simulations (horizontal dashed lines). Bottom panel: Fraction of voxels predicted to be collapsed (equivalent to $1-\beta$) as a function of the semantic threshold employed (solid black line); the horizontal black dashed line corresponds to the fraction of particles that end up in DM haloes measured in the validation simulations. In both panels, the vertical black dotted line shows the semantic threshold we employ; this threshold has been selected to match the fraction of collapsed voxels.

APPENDIX D: GENERATE FULL-BOX PREDICTIONS FROM CROPS

In this appendix, we address the challenge of generating full-box predictions employing our instance segmentation model.

While our network architecture captures intricate features within simulation sub-volumes, the challenge arises when we aim to apply it to arbitrarily large input domains. Unlike some other ML approaches that rely on networks that are translational invariant, our model incorporates the Lagrangian positions of particles as input channels, making it dependent on the relative Lagrangian position. This design choice ensures that similar regions of the initial density field are mapped to distinct locations in the pseudo-space, allowing us to distinguish between separate structures, even if they are locally identical. However, this feature also presents a challenge when creating full-box predictions. Combining independent crop predictions straightforwardly may lead to inconsistencies due to the network's inherent non-translational invariance. To tackle this issue, we have developed a methodology for predicting sub-volumes independently and then merging these predictions to generate accurate full-box instance segmentation results.

To reduce the boundary effects that may result from such a method we employ the following strategy.

(i) We evaluate the instance network centered several times, centered on locations $\vec{q}_{ijk}$ that are arranged on a grid

$$\vec{q}_{ijk} = \begin{pmatrix} i \cdot n_{\rm off} \\ j \cdot n_{\rm off} \\ k \cdot n_{\rm off} \end{pmatrix}, \tag{D1}$$

where we choose an offset of $n_{\rm off} = 64$ voxels and (i, j, k) run so far that the whole periodic volume is covered – e.g. from 0 to 4 each for a 256^3 simulation box. The network's input in each case corresponds to the 144^3 voxels (periodically) centered on $\vec{q}_{ijk}$ and the instance segmentation output will predict labels for the 128^3 central voxels.

(ii) From each prediction we only use the predicted labels of the central $n_{\rm off}^3 = 64^3$ voxels, since we expect these to be relatively robust to field-of-view effects. We combine these from all the predictions to a global grid that has the same dimensions as the input domain. In this step we add offsets to the labels so that the labels that originate from each predicted domain are unique in the global grid (this process will become relevant in step (iv) where we define a graph used to link instances).

(iii) We repeat steps (i)-(ii), but with an additional offset of $(n_{\rm off}/2, n_{\rm off}/2, n_{\rm off}/2)^T$. We additionally offset the labels in this second grid so that no label appears in both grids.

(iv) We use the two lattices and the intersections between instances to identify which labels should correspond to the same object. We do this by creating a graph[4] where each instance label is a node. Initially the graph has no edges, but we subsequently add edges if two labels should be identified (i.e. correspond to the same halo). Each connected component of the graph will then correspond to a single final label. To define the edges of the graph, we consider each quadrant Q of size $(n_{\rm off}/2)^3$ individually, since such quadrants are the maximal volumes over which two labels can intersect. We define the intersection $I_Q(l_1, l_2)$ of two labels l_1 and l_2 as the number of voxels that both carry label l_1 in grid one and label l_2 in grid two. We define as the union $U_Q(l_1, l_2)$ the number of voxels inside of quadrant Q that carry l_1 in grid 1 or l_2 in grid 2 (or both). We then add an edge between l_1 and l_2 into the graph if for any quadrant Q it is

$$\frac{I_Q(l_1, l_2)}{U_Q(l_1, l_2)} \geq IoU_{\rm thresh} \tag{D2}$$

where we set $IoU_{\rm thresh} = 0.5$.

(v) We summarize each connected component in the graph into a new label. After this operation for most voxels the new label in grid 1 and in grid 2 agree and we can choose that label as our final label. However, for a small fraction of voxels the labels still disagree, because the corresponding instances had too little overlap to be identified with each other. In this case we assign to the corresponding voxel the label that contains the larger number of voxels in total.

We illustrate the different steps of this procedure in Fig. D1. The top panel, labeled 'Lattice1', shows the individual instances predicted in the first lattice arrangement. Each color represents a distinct label assigned to a group of voxels within the 64^3 central region of the sub-volumes. The middle panel, 'Lattice2', displays the second set of predictions using a shifted lattice by half the offset in each dimension. Here again, different colors represent unique instance labels. The bottom panel, 'Combined', presents the final merged full-box prediction. It is generated by synthesizing the labels from 'Lattice1' and 'Lattice2' using the graph-based method to connect overlapping instances. The resulting image shows larger, coherent structures, indicative of the correct performance of combining both lattices.

Regarding the semantic segmentation network, we can merge the predictions corresponding to different crops independently since, in this case, we are truly working with a translation-invariant network. We employ the central 64^3 voxels (analogous to 'Lattice1') of separate predictions and merge them together to generate the final full-box predictions of the semantic segmentation network.

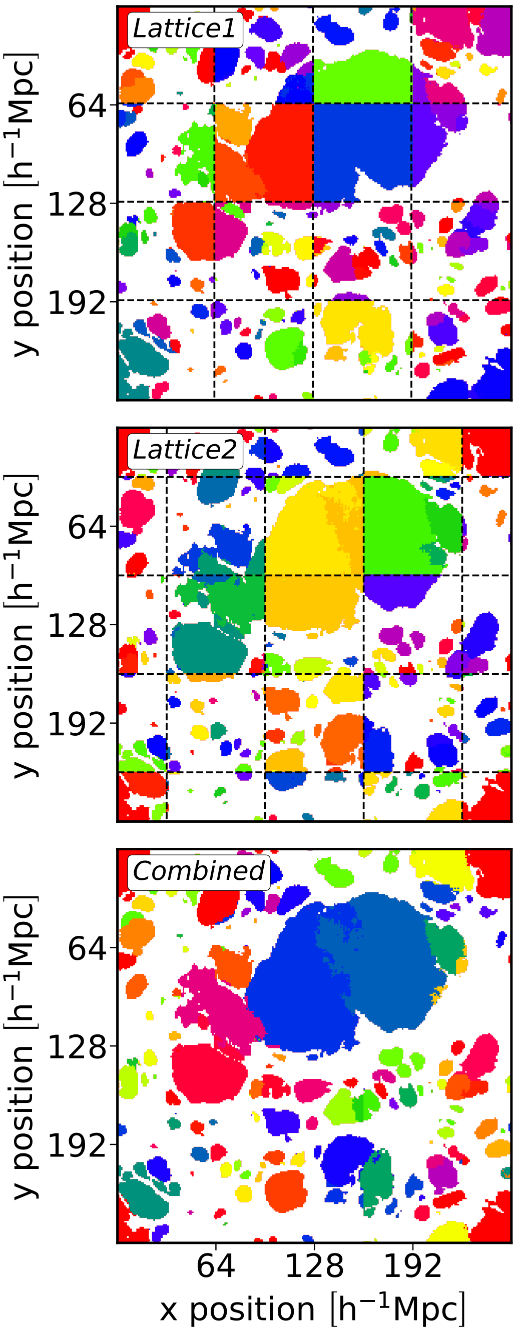

Figure D1. Process of merging predictions from two overlapping lattice structures to produce a full-box instance segmentation map. 'Lattice1' (**top**) and 'Lattice2' (**middle**) represent predictions from initial and shifted lattice grids, respectively, with unique color-coded labels for instances. 'Combined' (**bottom**) depicts the final synthesized full-box map, where instances have been merged based on their overlap, demonstrating the effectiveness of the methodology in generating contiguous and comprehensive halo segmentations from smaller, predicted sub-volumes.

This paper has been typeset from a TEX/LATEX file prepared by the author.

[4] using the NETWORKX library (Hagberg et al. 2008)