




# A Deep Learning Approach to Galaxy Cluster X-Ray Masses

M. Ntampaka<sup>1,2</sup> , J. ZuHone<sup>3</sup>, D. Eisenstein<sup>1</sup>, D. Nagai<sup>4</sup> , A. Vikhlinin<sup>1,5</sup> , L. Hernquist<sup>1</sup> , F. Marinacci<sup>1</sup>, D. Nelson<sup>6</sup>,  
R. Pakmor<sup>6</sup>, A. Pillepich<sup>7</sup> , P. Torrey<sup>8</sup>, and M. Vogelsberger<sup>9</sup>

<sup>1</sup> Center for Astrophysics | Harvard & Smithsonian, Cambridge, MA 02138, USA; [michelle.ntampaka@cfa.harvard.edu](mailto:michelle.ntampaka@cfa.harvard.edu)

<sup>2</sup> Harvard Data Science Initiative, Harvard University, Cambridge, MA 02138, USA

<sup>3</sup> Smithsonian Astrophysical Observatory, Cambridge, MA 02138, USA

<sup>4</sup> Department of Physics, Yale University, New Haven, CT 06520, USA

<sup>5</sup> Space Research Institute (IKI), Profsoyuznaya 84/32, Moscow, Russia

<sup>6</sup> Max-Planck-Institut für Astrophysik, Karl-Schwarzschild-Straße 1, D-85741, Garching bei München, Germany

<sup>7</sup> Max-Planck-Institut für Astronomie, Königstuhl 17, D-69117, Heidelberg, Germany

<sup>8</sup> Department of Astronomy, University of Florida, 211 Bryant Space Sciences Center, Gainesville, FL 32611, USA

<sup>9</sup> Kavli Institute for Astrophysics and Space Research, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Received 2018 October 17; revised 2019 March 27; accepted 2019 March 29; published 2019 May 7

## Abstract

We present a machine-learning (ML) approach for estimating galaxy cluster masses from *Chandra* mock images. We utilize a Convolutional Neural Network (CNN), a deep ML tool commonly used in image recognition tasks. The CNN is trained and tested on our sample of 7896 *Chandra* X-ray mock observations, which are based on 329 massive clusters from the *IllustrisTNG* simulation. Our CNN learns from a low resolution spatial distribution of photon counts and does not use spectral information. Despite our simplifying assumption to neglect spectral information, the resulting mass values estimated by the CNN exhibit small bias in comparison to the true masses of the simulated clusters ( $-0.02$  dex) and reproduce the cluster masses with low intrinsic scatter, 8% in our best fold and 12% averaging over all. In contrast, a more standard core-excised luminosity method achieves 15%–18% scatter. We interpret the results with an approach inspired by Google DeepDream and find that the CNN ignores the central regions of clusters, which are known to have high scatter with mass.

**Key words:** galaxies: clusters: general – methods: statistical – X-rays: galaxies: clusters

## 1. Introduction

Galaxy clusters are gravitationally bound systems that contain hundreds or thousands of galaxies in dark matter halos of mass  $\gtrsim 10^{14} M_{\odot}$ . They are massive and rare, and their abundance is sensitive to the underlying cosmological model. Utilizing cluster abundance as a cosmological probe requires a large cluster sample with a well-defined selection function, a way to connect observations to the underlying dark matter, and an understanding of the scatter in the mass-observable relationship.

A number of mass proxies can be derived from X-ray observations of galaxy clusters. X-ray cluster observations probe a portion of the baryonic component of clusters—the hot intracluster medium—which emits X-ray radiation primarily through bremsstrahlung. Hydrodynamical simulations (Nelson et al. 2014b; Hahn & Angulo 2016; Le Brun et al. 2017; Barnes et al. 2018; McCarthy et al. 2018) can be used to model observable-mass relations in clusters.

Cluster luminosity ( $L_X$ ) is correlated with mass, and excising the inner  $\approx 0.15 R_{500c}$  reduces scatter further due to variations in the core properties (Maughan 2007; Mantz et al. 2018). The global temperature ( $kT$ ) relates to cluster mass through the virial theorem, scaling with mass as a power law (e.g., Arnaud et al. 2005). For long exposures of bright, low-redshift clusters, it is possible to access luminosity and spectral cluster profiles, leading to tighter mass-observable relationships. Hydrostatic mass estimates can be calculated from temperature and density gradients (Vikhlinin et al. 2005) and the product of spectral temperature ( $T_X$ ) and gas mass ( $M_g$ ) denoted  $Y_X$ , is a very low-scatter mass proxy, with intrinsic scatter of  $\approx 5\%$ – $7\%$  (Kravtsov et al. 2006). However, the hydrostatic mass estimates are known to be biased (e.g., Nagai et al. 2007),

and it is one of the primary sources of systematic uncertainties in the cluster-based cosmological measurements (Planck Collaboration et al. 2016b).

A number of other physical processes impact X-ray based cluster mass estimates, including nonthermal pressure (Lau et al. 2009; Nelson et al. 2014a), gas clumping (Nagai & Lau 2011), temperature inhomogeneities (Rasia et al. 2014), and cluster dynamical state (Ventimiglia et al. 2008; Marrone et al. 2012). Calculable morphological parameters correlate with dynamical state, including surface brightness concentration (e.g., Santos et al. 2008), centroid shift (e.g., Rossetti et al. 2016), and morphological composite parameters (e.g., Rasia et al. 2013). These suggest that cluster observables are tied to mass in a complex way that may be exploited to reduce scatter and improve individual cluster mass estimates.

In addition to affecting mass estimate errors, a cluster’s dynamical state influences the probability that the cluster will be observed. Sunyaev–Zeldovich (SZ; Sunyaev & Zeldovich 1972)—selected samples preferentially have more disturbed clusters (Planck Collaboration et al. 2011), while X-ray-selected samples have a higher fraction of relaxed clusters (Eckert et al. 2011). If the fractions of relaxed and disturbed systems in cluster samples are not well known, this may introduce a bias (Randall et al. 2002).

Machine learning (ML) offers a number of tools that can be used to untangle subtle signals and extract complicated correlations. ML has been utilized in astronomy and cosmology for classification tasks such as labeling galaxy morphology (Banerji et al. 2010; Dieleman et al. 2015; Domínguez Sánchez et al. 2018), identifying transient types (Goldstein et al. 2015), identifying the presence or absence of lensing signals in images (Lanusse et al. 2018), categorizing the type of sources driving

reionization (Hassan et al. 2019), and estimating photometric redshifts (Pasquet et al. 2019). ML has also been used for astronomical and cosmological regression tasks, for example, reducing errors in cluster dynamical mass measurements (Ntampaka et al. 2015, 2016; Ho et al. 2019), determining the duration of reionization (La Plante & Ntampaka 2018), and producing tighter cosmological parameter constraints with mock catalogs (Gupta et al. 2018).

Because ML has been successful in harnessing complicated correlations in other astronomical applications, they may be useful in using subtle signals in X-ray images to improve mass estimates. One class of ML algorithms that has had much success in image-based tasks are Convolutional Neural Networks (CNNs; e.g., Fukushima & Miyake 1982; LeCun et al. 1999; Krizhevsky et al. 2012; Simonyan & Zisserman 2014). CNNs have many hidden layers, often pairing layers of convolution and pooling to extract features from the input images. They require very little preprocessing of the input images because the network learns the convolutional filters necessary to extract relevant features. See Schmidhuber (2015) for a review of deep neural networks.

We present a method for predicting cluster masses from decreased-resolution *Chandra* mock X-ray images that utilizes a CNN. We describe the mock observations in Section 2.1 and the CNN method and architecture in Section 2.2. We show the resulting mass predictions in Section 3, interpret the model in Section 4, and conclude in Section 5.

## 2. Methods

### 2.1. Mock Chandra Observations

#### 2.1.1. IllustrisTNG Clusters

A sample of simulated clusters is drawn from the *IllustrisTNG* cosmological hydrodynamical simulation (Marinacci et al. 2018; Naiman et al. 2018; Nelson et al. 2018; Pillepich et al. 2018a; Springel et al. 2018). *IllustrisTNG* uses an updated galaxy formation model (Weinberger et al. 2017; Pillepich et al. 2018c) to overcome many of the physical limitations of the previous *Illustris* model (Vogelsberger et al. 2013, 2014a, 2014b; Genel et al. 2014; Torrey et al. 2014; Nelson et al. 2015). The suite of *IllustrisTNG* simulations assumes a  $\Lambda$ CDM cosmology with parameters consistent with Planck Collaboration et al. (2016a). With a simulated cubic volume of 300 Mpc on a side, *TNG300* is the largest of the suite, making it ideal for studying rare and massive clusters. Furthermore, the simulation is performed at an unprecedented resolution, with baryonic mass resolution of  $7.6 \times 10^6 M_{\odot}$  (Nelson et al. 2018).

We select 329 massive clusters within a mass range of  $M_{500c} = 10^{13.57}$  to  $M_{500c} = 10^{15.06}$  from the *TNG300* simulation, using the Friends-of-Friends (FoF) “group” (Davis et al. 1985) halos from the  $z = 0$  snapshot. While an arbitrary spherical overdensity halo definition may include particles not linked within this FoF group, we are interested in predicting  $M_{500c}$  and the associated  $R_{500c}$  is small enough that all gas within this radius also should be found in the FoF group. Every gas cell associated with each group is included, so that all of the substructures associated with each cluster are used in the computation of the X-ray emission.

#### 2.1.2. pyXSIM

Our mock X-ray observations of the *IllustrisTNG* cluster sample are produced using the `pyXSIM`<sup>10</sup> (ZuHone et al. 2014) and `SOXS`<sup>11</sup> software packages. `pyXSIM` is an implementation of the PHOX algorithm (Biffi et al. 2012, 2013). Large photon samples are initially built in `pyXSIM` from the 3D distributions of density, temperature, and metallicity from the *IllustrisTNG* data for each cluster using an APEC emission model (Foster et al. 2012), assuming a redshift of  $z = 0.05$ . Only particles with  $kT > 0.1$  keV that are not forming stars are used in the construction of the photon samples. These samples are then projected along each of the  $x$ -,  $y$ -, and  $z$ -axes of the simulation box, and foreground galactic absorption is applied to each sample assuming the `wabs` (Morrison & McCammon 1983) model with a value of  $N_H = 4 \times 10^{20} \text{ cm}^{-2}$  for each cluster.

Each photon sample is then convolved with an instrument model for *Chandra*’s ACIS-I detector using the `SOXS` package. We assume a simplified representation of the ACIS-I detector, with a 20’ square field of view without chip gaps and 0’’5 pixels. At our cluster sample redshift,  $R_{500c}$  extends beyond the 20’ square field of view of the detector for clusters with mass  $M_{500c} \gtrsim 10^{13.8} M_{\odot}$ . Rather than producing a tiled *Chandra* image that fully contains  $R_{500c}$ , we opt to use mock cluster observations that could be achieved from a single pointing. The point-spread function (PSF) is Gaussian-shaped with FWHM 0’’5, and the effective area and spectral response are taken from the Cycle 19 aimpoint response files (ARF and RMF) and assumed to be the same across the entire detector. The built-in models for the ACIS-I particle background and the galactic foreground included with `SOXS` were also applied.<sup>12</sup> We integrate each observation for 100 ks.

#### 2.1.3. Image Preprocessing

Because the ML tool described in Section 2.2 is not invariant under image rotation,<sup>13</sup> we augment the data set with 90° rotations as well as reflections along the vertical and horizontal axes (as in, e.g., Cabrera-Vives et al. 2017). The three 2D projections, two axial reflections, and four possible 90° orientations result in 24 images for each unique cluster. All 24 images of each unique cluster are assigned to one of 10 groups, called folds. The clusters are ordered by mass and cyclically assigned to folds so that each fold has approximately the same mass function.

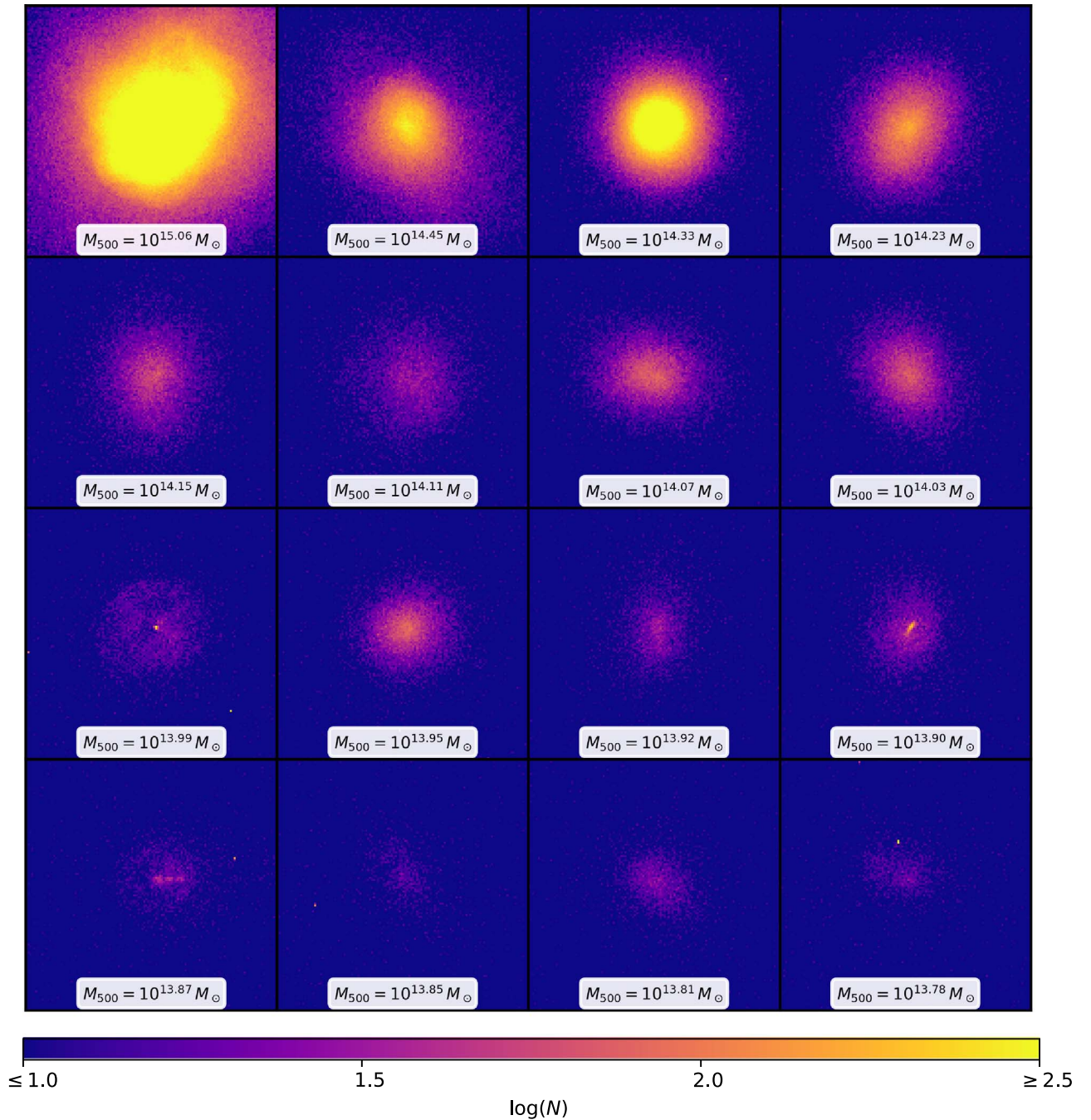
Each mock *Chandra* event file is degraded in spatial resolution to a  $128 \times 128$  “postage stamp” image over the broad energy band of 0.5–7 keV. The decreased resolution has two advantages: it decreases computation time and it also decreases the effects of the nonuniform distortion of the *Chandra* PSF. Moving to a lower resolution reduces the effects of the nonuniform PSF in a real observation, making it unnecessary to model it precisely. Example images of a representative sample of 16 clusters spanning the mass range are shown in Figure 1. The final catalog is comprised of 24 decreased-resolution *Chandra* images of each of 329 unique *IllustrisTNG* clusters, totaling 7896 *Chandra* mock observations.

<sup>10</sup> <http://hea-www.cfa.harvard.edu/~jzuhone/pyxsim/>

<sup>11</sup> <http://hea-www.cfa.harvard.edu/~jzuhone/soxs/>

<sup>12</sup> [http://hea-www.cfa.harvard.edu/~jzuhone/soxs/users\\_guide/background.html](http://hea-www.cfa.harvard.edu/~jzuhone/soxs/users_guide/background.html)

<sup>13</sup> Rotationally invariant CNNs are an area of active research, see, for example, Dieleman et al. (2015) and Worrall et al. (2017).



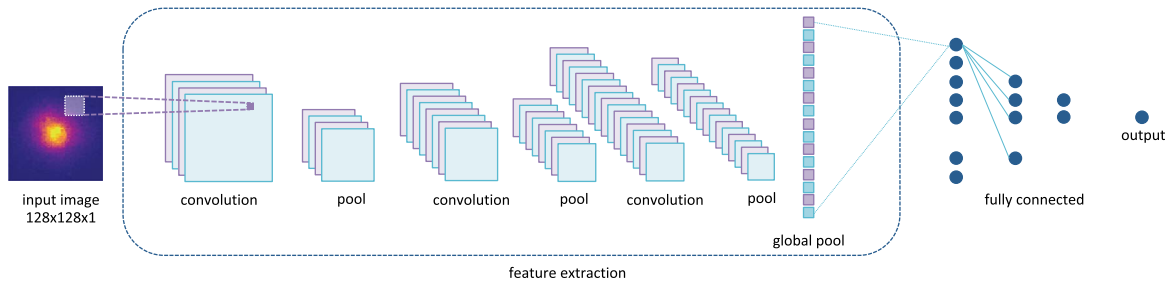
**Figure 1.** Sample of 16 of the 7896 mock X-ray cluster observations created with pyXSIM software applied to the *IllustrisTNG* cosmological hydrodynamical simulation. The mock observations emulate 100 ks *Chandra* observations that have been degraded to  $128 \times 128$  postage stamp images for one broad (0.5–7 keV) energy band; shown are the number of photons,  $N$ , in each pixel for this mock observation. Each unique cluster in the simulation is used to produce 24 mock images according to the data augmentation scheme described in Section 2.1.

## 2.2. Convolutional Neural Networks

CNNs (Fukushima & Miyake 1982; LeCun et al. 1999; Krizhevsky et al. 2012) are a class of feed-forward ML algorithms that are commonly used in image recognition tasks. They use pairs of convolutional filters and pooling layers to extract meaningful patterns from the input image, and can be used for both classification and regression tasks. Because the network learns the convolutional filters, CNNs require very little preprocessing of the input images.

The CNN is implemented in Keras (Chollet 2015) with a Tensorflow (Abadi et al. 2016) backend. Our CNN architecture is based loosely on a simplified version of Simonyan & Zisserman (2014) with fewer hidden layers; it is shown in Figure 2. The model is implemented with sequential layers as follows:

1.  $3 \times 3$  convolution with 16 filters.
2.  $2 \times 2$ , stride-2 max pooling.
3.  $3 \times 3$  convolution with 32 filters.



**Figure 2.** Architecture of the Convolutional Neural Network (CNN) used in this analysis. Our network utilizes three convolutional and pooling layers for feature extraction and three fully connected layers for parameter estimation.

4.  $2 \times 2$ , stride-2 max pooling.
5.  $3 \times 3$  convolution with 64 filters.
6.  $2 \times 2$ , stride-2 max pooling.
7. Global average pooling.
8. 10% dropout.
9. 200 neurons, fully connected.
10. 10% dropout.
11. 100 neurons, fully connected.
12. 20 neurons, fully connected.
13. Output neuron.

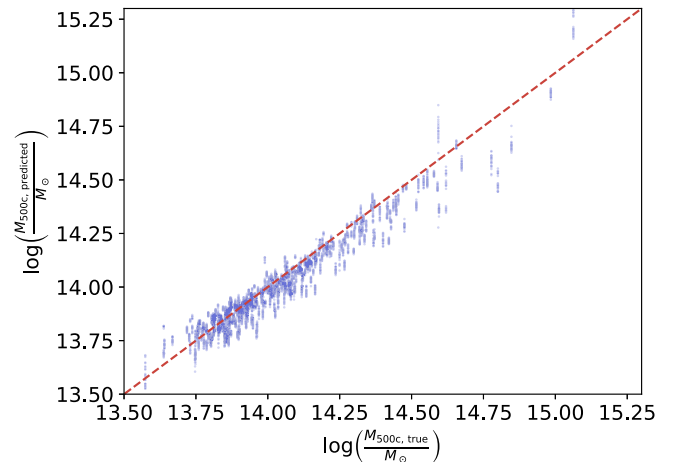
We use a root-mean-squared loss function and the Adam Optimizer (Kingma & Ba 2014) with learning rate reduced to half of the default value ( $\text{lr} = 0.0005$ ).

In our model, feature extraction is performed by three pairs of  $3 \times 3$  convolutional filters coupled with  $2 \times 2$  stride-2 max pooling layers (e.g., Riesenhuber & Poggio 1999). These are followed by a global average pooling layer (Lin et al. 2013) and three fully connected layers with rectified linear unit (ReLU, Nair & Hinton 2010) activation. A 10% dropout after two fully connected layers prevents overfitting (Srivastava et al. 2014).

For the task of regressing a single parameter, we use an architecture with one output neuron. This output neuron gives a continuous-valued label (regression) rather than a class probability (classification). We select the mean-squared loss function for this regression task. Our model has 58,437 tunable parameters; because of the global average pooling layer that compresses the information into 64 neurons, the number of tunable parameters is invariant to changes to the input image resolution.

We use the  $128 \times 128$  images as input and train the model to predict  $\log(M_{500c})$  from these images. We perform a 10-fold cross-validation, dividing the sample into a training set that comprises 80% of the images, a validation set that comprises 10% of the images, and a test set of the remaining 10% of the images. Every rotation, axial flip, and line-of-sight view of a single cluster is assigned to only one of these sets; a cluster is never used, for example, to train a model and subsequently test it. The training set is used to train the model to minimize a mean squared error loss function, the validation set is used to assess the stopping criteria, and the mass predictions for the test set are reported. We cycle through the data assignments to the train, validation, and test folds until the masses of all clusters have been predicted.

Stopping criteria are implemented to converge on models that, when applied to the validation set, are low scatter, low bias, and have no catastrophic outliers. The mass residual,  $\delta$ , is



**Figure 3.** Predicted mass as a function of true mass. The distribution has low intrinsic scatter (11.6%) and a small negative bias ( $-0.022$  dex). The tendency to predict toward the mean—overpredicting low mass clusters and underpredicting high mass clusters—can be mitigated by carefully curating a training set that extends well beyond the mass range of test clusters.

defined as

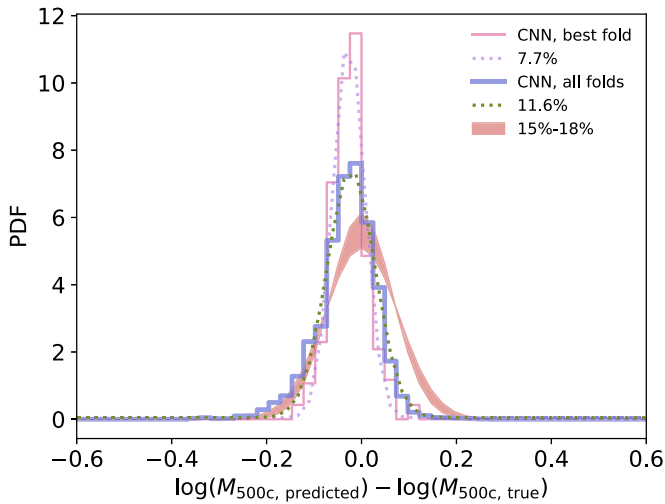
$$\delta \equiv \log(M_{\text{predicted}}) - \log(M_{\text{true}}). \quad (1)$$

We consider the training of the CNN to have converged when the following three criteria are met: maximum absolute value of mass residual is less than 0.3, absolute value of median residual error is less than 0.02, slope of the best-fit line of  $M_{\text{true}}$  versus  $M_{\text{predicted}}$  is greater than 0.9. For several folds, these three criteria were not met, so the model at the 400th training cycle, or “epoch,” is used. In most cases, the training converged within 150 epochs. We emphasize that these criteria are applied to the validation set and these criteria may not be descriptive of the test set.

### 3. Results

The mass predictions for all 10 folds are shown in Figure 3. Vertical streams of points show the mass estimates for each of the 24 images of each cluster. The results show a tendency to predict toward the mean, evident in the overprediction of low mass clusters and the underprediction of high mass clusters. When applying this method to a sample of observed clusters, one would create a training catalog that extends beyond the estimated mass range of the test catalog to mitigate this issue.

A PDF of mass errors is shown in Figure 4. The 10-fold distribution is fit to a Gaussian with width  $\sigma = 0.051$  dex (corresponding to a 11.6% scatter) and a small negative bias given by the mean  $\mu = -0.022$  dex. In practice, the train set



**Figure 4.** PDF of mass error (blue solid) given by  $\log(M_{\text{predicted}}) - \log(M_{\text{true}})$ . The full sample error distribution best-fit Gaussian (green dash) has standard deviation  $\sigma = 0.051$  dex (11.6% intrinsic scatter) and mean  $\mu = -0.022$  dex. The best-fit fold has a width  $\sigma = 0.033$  dex (7.7% intrinsic scatter) and mean  $\mu = -0.027$  dex (pink solid and purple dash). Core-excised  $L_X$ -based methods that use a single measure of cluster luminosity typically achieve a 15%–18% scatter (red band), while the  $Y_X$  technique, which requires the full spatial and spectral observation, can yield a tighter 5%–7% scatter. Our CNN approach uses low resolution spatial information to improve mass estimates over one based on a single summary parameter,  $L_X$ .

should extend well beyond the estimated mass range of test clusters, using the model to make predictions for clusters at the middle mass region to mitigate the effects of bias to the mean. In the mass range  $14.0 < \log(M_{500c}) < 14.5$ , the 10-fold distribution is well-described by a Gaussian with width  $\sigma = 0.41$  dex, 9.6% scatter, and bias  $\mu = -0.036$  dex.

The initial random state of a CNN can affect the solution upon which it converges, and so it is informative not only to evaluate the 10-fold mean, but also to evaluate the best-fit fold. For the full mass range, the best-fit fold has a width  $\sigma = 0.033$  dex, 7.7% scatter, and bias  $\mu = -0.027$  dex. In the mass range  $14.0 < \log(M_{500c}) < 14.5$ , the best-fit fold has a width  $\sigma = 0.025$  dex, 5.7% scatter, and bias  $\mu = -0.021$  dex.

Putting this intrinsic scatter in context, luminosity ( $L_X$ )-based methods with excised cores typically have errors in the 15%–18% range (Maughan 2007; Mantz et al. 2018), while methods that utilize well-sampled clusters with high spatial and spectral resolution, such as a  $Y_X$  approach, yield 5%–7% intrinsic scatter (Kravtsov et al. 2006). A straightforward power-law scaling relation relating mass to core-excised luminosity of the *IllustrisTNG* cluster sample recovers the approximate expected scatter: 14.6% when the outer aperture has a modest 3% error with  $R_{500}$ , and 22.2% when the outer aperture has a 5% error with  $R_{500}$ . See A. Pop et al. (2019, in preparation), for more information on the *IllustrisTNG* cluster sample scaling relations.

Here, we have used low resolution spatial information with no spectral data and have achieved a  $>20\%$  improvement over a global luminosity approach. In practice, a single, low-scatter model could be selected for an application of this method, implying that an improvement closer to 50% is possible.

As larger cosmological hydrodynamical simulations with more massive clusters become available, the scatter and bias of mass predictions may also be reduced by training on a

cluster sample with a flat mass function (as is used in Ntampaka et al. 2015) that more accurately describes the high mass cluster population.

#### 4. Interpreting the Model with DeepDream

CNNs are notoriously difficult to interpret. To understand the method, we use an approach inspired by DeepDream.<sup>14</sup> Google DeepDream uses gradient ascent applied to the input image pixels, asking the question “What changes in the input image will result in a significant change in the classification of this image?” Often, DeepDream is used to change the classification of a picture or photograph. Our implementation differs in that our model regresses an output mass label, so we will be asking “What changes in the input cluster image will result in a mass change of this image?”<sup>15</sup>

We define our loss function as the value of the final neuron output (the cluster mass) and compute the gradient of the input image with respect to this loss. The gradient is a  $128 \times 128$  single-color image and is calculated by finding the changes needed in each of the 16,384 pixels to maximize the cluster mass. It is the change to the input image that maximizes the loss function, in other words, adding the gradient to the input image results in a cluster that the trained CNN model interprets as being more massive.

Adding the image gradient can introduce nonphysical properties to the image, including pixels with noninteger and negative photon counts. To correct for this, we impose physically motivated constraints on the input image plus gradient: pixels with negative photon counts are set to 0 and pixels with noninteger values are rounded to the nearest integer. It should be noted that these physically motivated constraints do not significantly affect the CNN’s prediction of the new cluster mass, nor do they significantly affect the plots or results presented here.

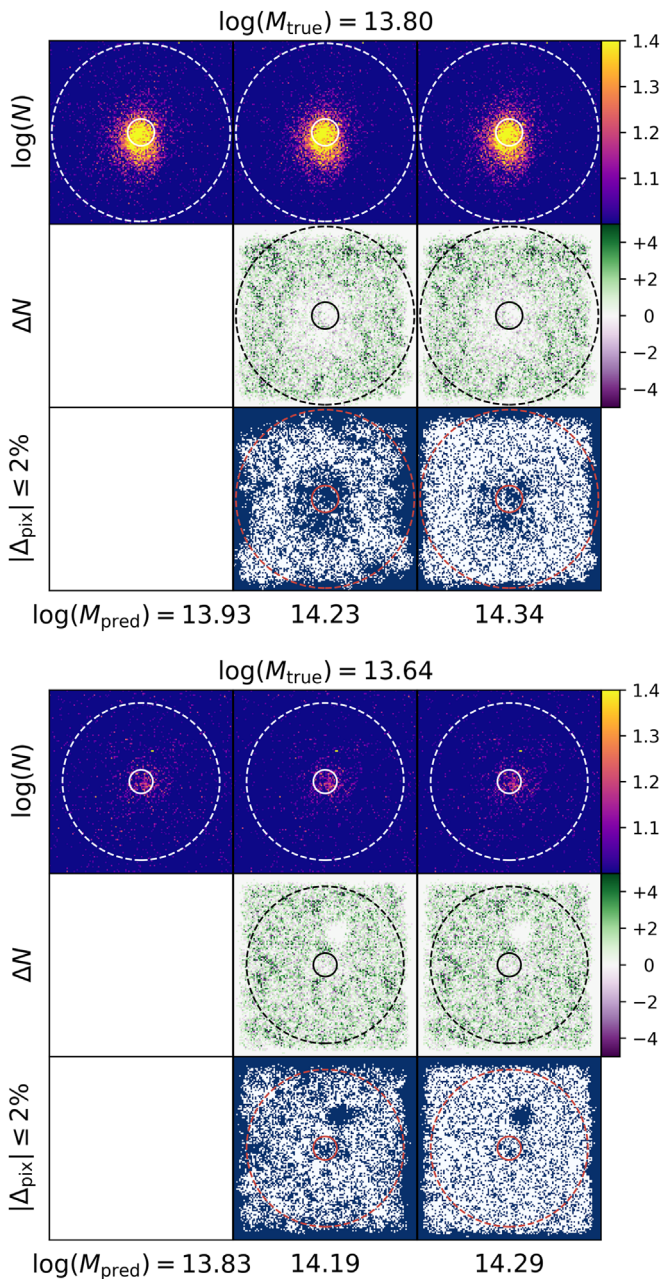
Figure 5 shows two sample clusters, including the input image, the gradients, and the updated cluster images for two iterations of this process. The trained model typically adds photons outside of  $\approx 0.2 R_{500c}$  but ignores the core region of the cluster that is known to have large scatter with cluster mass (Maughan 2007; Mantz et al. 2018).

Figure 6 shows the fractional photon change,  $\Delta N/N$ , for a representative sample of clusters. This is given by the ratio of photons in the iterated image (original image plus gradient) to the photons in the original image. The two-dimensional result is binned by radius. The CNN ignores the central  $\approx 0.2 R_{500c}$  of the cluster, typically adding photons outside of this region. In the second iteration, photons are added even further from the cluster center. This tool tends not to add photons near the edge of images, suggesting that some edge effects come into play.

One notable exception is shown in the bottom panel of Figure 5 and highlighted in Figure 6. In this case, the CNN adds photons near the core region. Further inspection of this cluster reveals two bright, off-center pixels above and to the right of the cluster center. The CNN incorrectly interprets this bright region as the cluster core, perturbing the image by adding photons surrounding it. These bright photons likely originated from nonphysical, high-density, ionized cold

<sup>14</sup> <https://ai.googleblog.com/2015/07/deepdream-code-example-for-visualizing.html>

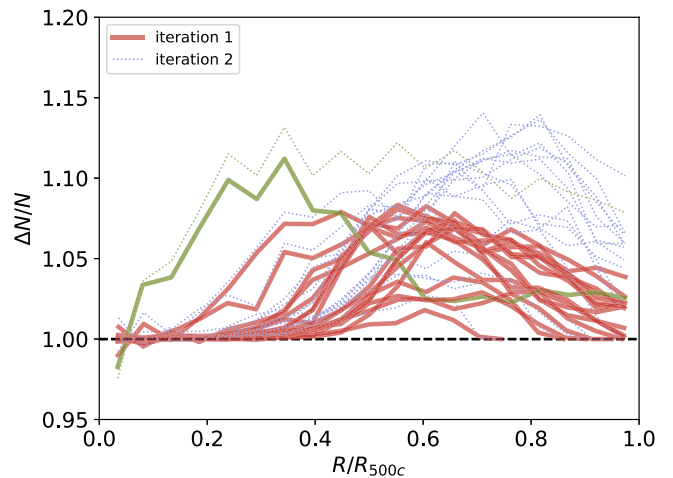
<sup>15</sup> For more details on visualizing filters of CNNs implemented in Keras, see <https://blog.keras.io/how-convolutional-neural-networks-see-the-world.html>.



**Figure 5.** Top panel: a typical cluster’s evolution over two iterations in the DeepDream-inspired tool for interpreting the CNN. Top row: the original input image ( $\log(N)$ ) of the cluster (left) is perturbed through two iterations (center and right) to increase the apparent cluster mass. Middle row: changes in photon count ( $\Delta N$ ) for each iteration (center and right) shows that the CNN tends to add photons roughly in a ring between  $0.15 R_{500c}$  (solid circle) and  $1.0 R_{500c}$  (dashed circle). Bottom row: dark pixels show the regions for which there is a small photon change ( $\Delta N/N \leq 2\%$ ). Bottom panel: same as top panel, but for one notable cluster for which the CNN misidentifies the cluster core above and to the right of the true cluster core. This is highlighted in the bottom right image, where an off-center circular region has small photon count change.

particles that are not easily filtered out. In analysis of real *Chandra* observations, point sources like these would be removed in preprocessing. When applying the CNN mass estimate method to a sample of observed clusters, our interpretation may be useful in identifying, categorizing, and understanding outliers such as this cluster.

Our data-driven model, calibrated on a simulation, has achieved small bias and scatter without invoking assumptions



**Figure 6.** Fractional change in photons ( $\Delta N/N$ ) as a function of projected distance from the cluster center ( $R/R_{500c}$ ) for a representative sample of clusters for the DeepDream interpretation of the trained CNN. The first iteration (solid red) adds photons beyond  $\approx 0.2 R_{500c}$ , while the second iteration (blue dotted) increases the photon count at larger radii. This suggests that the CNN has learned to excise cores, which have been shown to have large scatter with  $M_{500c}$ . One notable exception (green solid and dotted) is the cluster highlighted in the bottom panel of Figure 5. The gradients for this cluster have an empty region near the cluster’s off-center bright region; this cluster is discussed in more detail in Section 4. Interpretation tools such as the one presented here can be used to understand the features used by a CNN to regress cluster mass.

about hydrostatic equilibrium or hydrostatic mass bias. The model has learned to excise cores, ignoring the region of the cluster most directly affected by feedback physics, which is difficult to model. Although the global intercluster medium profiles are relatively unaffected by still poorly understood cluster core physics (Sembolini et al. 2016), *IllustrisTNG*-calibrated mass estimates will depend on the physics of cluster outskirts (see Walker et al. 2019 for a recent review). Encouragingly, *IllustrisTNG* clusters recover the expected X-ray scaling relations (A. Pop et al. 2019, in preparation) and metallicity profiles (Vogelsberger et al. 2018), suggesting that the outskirts of these simulated clusters are in good agreement with observed clusters. To confirm that no significant bias is introduced by the gas physics modeling, weak lensing mass estimates of a well-studied cluster sample can provide an important cross-check.

## 5. Conclusion

We have presented a method for inferring cluster masses from *Chandra* mock observations of galaxy clusters. The mock observations are built from 329 massive clusters within a mass range of  $M_{500c} = 10^{13.57}$  to  $M_{500c} = 10^{15.06}$  from the *TNG300* simulation. The mass proxy uses a CNN with three pairs of convolutional and pooling layers followed by three fully connected layers. The model is trained to learn cluster mass from a low spatial resolution, single-color X-ray image. Our approach shows that a low resolution X-ray image of a galaxy cluster can be used to predict the mass with low scatter (12%) and low bias ( $-0.02$  dex) without invoking assumptions about the hydrostatic mass bias.

The scatter may improve as larger simulations become available, providing catalogs that better sample the high mass region of the halo mass function. Ultimately, this method may

be trained on simulations to predict the masses of *Chandra*-observed clusters.

ML tools are commonly viewed as black boxes that produce an answer without an interpretation, and it can be particularly difficult to glean a physical understanding from deep learning methods. We aim to remedy this by studying our trained model with an approach inspired by Google DeepDream. We calculate a gradient necessary to perturb an input cluster image so that the trained CNN will increase the mass estimate. We find that the trained CNN is most sensitive to photons from the cluster outskirts and ignores the inner ( $R \lesssim 0.2R_{500c}$ ) regions of the cluster, in agreement with what has been found by more conventional statistical analyses of galaxy clusters. The method can be useful in providing a physical interpretation of the features of the cluster sample that are relevant for predicting masses from X-ray images.

As new, large cluster samples become available, new data-driven methods will need to be developed to take advantage of these data sets. For example, the upcoming *eROSITA* mission (Merloni et al. 2012) is estimated to find  $\approx 93,000$  galaxy clusters with masses larger than  $10^{13.7} h^{-1} M_{\odot}$  (Pillepich et al. 2012, 2018b). New tools, such as the CNN-based mass proxy presented here, can be useful in analyzing and understanding large observational data sets. Utilizing CNNs to infer X-ray masses of the *eROSITA* cluster sample, however, will require a much larger training sample of simulated images spanning a wide dynamic range in cluster mass. As bigger simulated cluster catalogs become available, CNNs may prove to be a powerful tool for analyzing and understanding large cluster observations.

We thank Dominique Eckert, Melanie Fernandez, Sheridan Green, François Lanusse, Paul La Plante, Juneri Oliva, Kun-Hsing Yu, and Javier Zazo for their helpful feedback on this project.

### ORCID iDs

M. Ntampaka  <https://orcid.org/0000-0002-0144-387X>  
 D. Nagai  <https://orcid.org/0000-0002-6766-5942>  
 A. Vikhlinin  <https://orcid.org/0000-0001-8121-0234>  
 L. Hernquist  <https://orcid.org/0000-0001-6950-1629>  
 A. Pillepich  <https://orcid.org/0000-0003-1065-9274>

### References

- Abadi, M., Barham, P., Chen, J., et al. 2016, in 12th USENIX Symp. Operating Systems Design and Implementation (OSDI 16) (Berkeley, CA: USENIX), 265, <https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf>
- Arnaud, M., Pointecouteau, E., & Pratt, G. W. 2005, *A&A*, 441, 893
- Banerji, M., Lahav, O., Lintott, C. J., et al. 2010, *MNRAS*, 406, 342
- Barnes, L. A., Elahi, P. J., Salcido, J., et al. 2018, *MNRAS*, 477, 3727
- Biffi, V., Dolag, K., & Böhringer, H. 2013, *MNRAS*, 428, 1395
- Biffi, V., Dolag, K., Böhringer, H., & Lemson, G. 2012, *MNRAS*, 420, 3545
- Cabrera-Vives, G., Reyes, I., Förster, F., Estévez, P. A., & Maureira, J.-C. 2017, *ApJ*, 836, 97
- Chollet, F. 2015, keras, <https://github.com/fchollet/keras>
- Davis, M., Efstathiou, G., Frenk, C. S., & White, S. D. M. 1985, *ApJ*, 292, 371
- Dieleman, S., Willett, K. W., & Dambre, J. 2015, *MNRAS*, 450, 1441
- Domínguez Sánchez, H., Huertas-Company, M., Bernardi, M., Tuccillo, D., & Fischer, J. L. 2018, *MNRAS*, 476, 3661
- Eckert, D., Molendi, S., & Paltani, S. 2011, *A&A*, 526, A79
- Foster, A. R., Ji, L., Smith, R. K., & Brickhouse, N. S. 2012, *ApJ*, 756, 128
- Fukushima, K., & Miyake, S. 1982, in Competition and Cooperation in Neural Nets (Berlin: Springer), 267
- Genel, S., Vogelsberger, M., Springel, V., et al. 2014, *MNRAS*, 445, 175
- Goldstein, D. A., D'Andrea, C. B., Fischer, J. A., et al. 2015, *AJ*, 150, 82
- Gupta, A., Matilla, J. M. Z., Hsu, D., & Haiman, Z. 2018, *PhRvD*, 97, 103515
- Hahn, O., & Angulo, R. E. 2016, *MNRAS*, 455, 1115
- Hassan, S., Liu, A., Kohn, S., & La Plante, P. 2019, *MNRAS*, 483, 2524
- Ho, M., Rau, M. M., Ntampaka, M., et al. 2019, *ApJ*, submitted arXiv:1902.05950
- Kingma, D. P., & Ba, J. 2014, arXiv:1412.6980
- Kravtsov, A. V., Vikhlinin, A., & Nagai, D. 2006, *ApJ*, 650, 128
- Krizhevsky, A., Sutskever, I., & Hinton, G. E. 2012, in Advances in Neural Information Processing Systems 25, ed. F. Pereira et al. (Red Hook, NY: Curran Associates Inc.), 1097
- Lanusse, F., Ma, Q., Li, N., et al. 2018, *MNRAS*, 473, 3895
- La Plante, P., & Ntampaka, M. 2018, *ApJ*, submitted arXiv:1810.08211
- Lau, E. T., Kravtsov, A. V., & Nagai, D. 2009, *ApJ*, 705, 1129
- Le Brun, A. M. C., McCarthy, I. G., Schaye, J., & Ponman, T. J. 2017, *MNRAS*, 466, 4442
- LeCun, Y., Haffner, P., Bottou, L., & Bengio, Y. 1999, in Shape, Contour and Grouping in Computer Vision (Berlin: Springer), 319
- Lin, M., Chen, Q., & Yan, S. 2013, arXiv:1312.4400
- Mantz, A. B., Allen, S. W., Morris, R. G., & von der Linden, A. 2018, *MNRAS*, 473, 3072
- Marinacci, F., Vogelsberger, M., Pakmor, R., et al. 2018, *MNRAS*, 480, 5113
- Marrone, D. P., Smith, G. P., Okabe, N., et al. 2012, *ApJ*, 754, 119
- Maughan, B. J. 2007, *ApJ*, 668, 772
- McCarthy, I. G., Bird, S., Schaye, J., et al. 2018, *MNRAS*, 476, 2999
- Merloni, A., Predehl, P., Becker, W., et al. 2012, arXiv:1209.3114
- Morrison, R., & McCammon, D. 1983, *ApJ*, 270, 119
- Nagai, D., & Lau, E. T. 2011, *ApJL*, 731, L10
- Nagai, D., Vikhlinin, A., & Kravtsov, A. V. 2007, *ApJ*, 655, 98
- Naiman, J. P., Pillepich, A., Springel, V., et al. 2018, *MNRAS*, 477, 1206
- Nair, V., & Hinton, G. E. 2010, in Proc. 27th Int. Conf. Machine Learning 10 (ICML-10), ed. J. Fürnkranz & T. Joachims (Madison, WI: Omnipress), 807, <https://icml.cc/Conferences/2010/papers/432.pdf>
- Nelson, D., Pillepich, A., Genel, S., et al. 2015, *A&C*, 13, 12
- Nelson, D., Pillepich, A., Springel, V., et al. 2018, *MNRAS*, 475, 624
- Nelson, K., Lau, E. T., & Nagai, D. 2014a, *ApJ*, 792, 25
- Nelson, K., Lau, E. T., Nagai, D., Rudd, D. H., & Yu, L. 2014b, *ApJ*, 782, 107
- Ntampaka, M., Trac, H., Sutherland, D. J., et al. 2015, *ApJ*, 803, 50
- Ntampaka, M., Trac, H., Sutherland, D. J., et al. 2016, *ApJ*, 831, 135
- Pasquet, J., Bertin, E., Treyer, M., Arnouts, S., & Fouchez, D. 2019, *A&A*, 621, A26
- Pillepich, A., Nelson, D., Hernquist, L., et al. 2018a, *MNRAS*, 475, 648
- Pillepich, A., Porciani, C., & Reiprich, T. H. 2012, *MNRAS*, 422, 44
- Pillepich, A., Reiprich, T. H., Porciani, C., Borm, K., & Merloni, A. 2018b, *MNRAS*, 481, 613
- Pillepich, A., Springel, V., Nelson, D., et al. 2018c, *MNRAS*, 473, 4077
- Planck Collaboration, Ade, P. A. R., Aghanim, N., et al. 2016a, *A&A*, 594, A13
- Planck Collaboration, Ade, P. A. R., Aghanim, N., et al. 2016b, *A&A*, 594, A24
- Planck Collaboration, Aghanim, N., Arnaud, M., et al. 2011, *A&A*, 536, A9
- Randall, S. W., Sarazin, C. L., & Ricker, P. M. 2002, *ApJ*, 577, 579
- Rasia, E., Lau, E. T., Borgani, S., et al. 2014, *ApJ*, 791, 96
- Rasia, E., Meneghetti, M., & Ettori, S. 2013, *AstRv*, 8, 40
- Riesenhuber, M., & Poggio, T. 1999, *Nat. Neurosci.*, 2, 1019
- Rossetti, M., Gastaldello, F., Ferioli, G., et al. 2016, *MNRAS*, 457, 4515
- Santos, J. S., Rosati, P., Tozzi, P., et al. 2008, *A&A*, 483, 35
- Schmidhuber, J. 2015, *Neural Networks*, 61, 85
- Sembolini, F., Elahi, P. J., Pearce, F. R., et al. 2016, *MNRAS*, 459, 2973
- Simonyan, K., & Zisserman, A. 2014, arXiv:1409.1556
- Springel, V., Pakmor, R., Pillepich, A., et al. 2018, *MNRAS*, 475, 676
- Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. 2014, *J. Mach. Learn. Res.*, 15, 1929
- Sunyaev, R. A., & Zeldovich, Y. B. 1972, *CoASP*, 4, 173
- Torrey, P., Vogelsberger, M., Genel, S., et al. 2014, *MNRAS*, 438, 1985
- Ventimiglia, D. A., Voit, G. M., Donahue, M., & Ameglio, S. 2008, *ApJ*, 685, 118
- Vikhlinin, A., Markevitch, M., Murray, S. S., et al. 2005, *ApJ*, 628, 655
- Vogelsberger, M., Genel, S., Sijacki, D., et al. 2013, *MNRAS*, 436, 3031
- Vogelsberger, M., Genel, S., Springel, V., et al. 2014a, *MNRAS*, 444, 1518
- Vogelsberger, M., Genel, S., Springel, V., et al. 2014b, *Natur*, 509, 177
- Vogelsberger, M., Marinacci, F., Torrey, P., et al. 2018, *MNRAS*, 474, 2073
- Walker, S., Simionescu, A., Nagai, D., et al. 2019, *SSRv*, 215, 7
- Weinberger, R., Springel, V., Hernquist, L., et al. 2017, *MNRAS*, 465, 3291
- Worrall, D. E., Garbin, S. J., Turmukhambetov, D., & Brostow, G. J. 2017, in 2017 IEEE Conf. on Computer Vision and Pattern Recognition (CVPR) (New York: IEEE), 7168
- ZuHone, J. A., Biffi, V., Hallman, E. J., et al. 2014, in Proc. 13th Python in Science Conf., ed. S. van der Walt & J. Bergstra, 103, <http://conference.scipy.org/proceedings/scipy2014/zuhone.html>