Draft version June 25, 2020
Typeset using LaTeX **twocolumn** style in AASTeX63

arXiv:2006.13231v1 [astro-ph.CO] 23 Jun 2020

Approximate Bayesian Uncertainties on Deep Learning Dynamical Mass Estimates of Galaxy Clusters

Matthew Ho,[1] Arya Farahi,[2] Markus Michael Rau,[1] and Hy Trac[1]

[1] *McWilliams Center for Cosmology, Department of Physics, Carnegie Mellon University, Pittsburgh, PA 15213, USA*
[2] *The Michigan Institute for Data Science, University of Michigan, Ann Arbor, MI 48109, USA*

Submitted to ApJ

ABSTRACT

We study methods for reconstructing Bayesian uncertainties on dynamical mass estimates of galaxy clusters using Convolutional Neural Networks (CNNs). We discuss the statistical background of Bayesian Neural Networks and demonstrate how variational inference techniques can be used to perform computationally tractable posterior estimation for a variety of deep neural architectures. We explore how various model designs and statistical assumptions impact prediction accuracy and uncertainty reconstruction in the context of cluster mass estimation. We measure the quality of our model posterior recovery using a mock cluster observation catalog derived from the MultiDark simulation and UniverseMachine catalog. We show that Bayesian CNNs produce highly accurate dynamical cluster mass posteriors. These model posteriors are log-normal in cluster mass and recover 68% and 90% confidence intervals to within 1% of their measured value. We note how this rigorous modelling of dynamical mass posteriors is necessary for using cluster abundance measurements to constrain cosmological parameters.

Keywords: cosmology: theory - galaxies: clusters: general - galaxies: kinematics and dynamics - methods: statistical

1. INTRODUCTION

Galaxy clusters are the most massive gravitationally bound systems in the universe, consisting of hundreds of luminous galaxies and hot gas embedded in dense dark matter halos. The distribution of cluster masses dominates the sensitive high mass regime of the halo mass function (HMF) and is a useful probe of large-scale structure. Measurements of cluster abundance as a function of halo mass and redshift are a major method for constraining cosmological models, but such analyses require large, well-defined cluster samples and robust mass measurement methods (e.g. Voit 2005; Allen et al. 2011; Mantz et al. 2015; Planck Collaboration et al. 2016). As the number of high-quality cluster observations is expected to radically increase with current and upcoming cosmological surveys such as the Dark Energy Spectroscopic Instrument (DESI), the Vera C. Rubin Observatory, and *Euclid* (Dodelson et al. 2016), the need for precise and efficient cluster mass estimators is imperative.

Dynamical mass estimators are a class of cluster measurements which leverage information from spectroscopic observations of member galaxies in order to infer cluster masses. The theoretical foundations of dynamical methods are grounded in the M-σ relation, a fundamental power-law relationship which connects the mass of a stable, isotropic cluster system to the line-of-sight (LOS) velocity dispersion of its constituent galaxies. Such methods were famously used to produce the first inference of the existence of dark matter in the Coma cluster (Zwicky 1933). Despite this historical significance, vanilla applications of the M-σ relation produce significant biases and scatter in realistic cluster mass predictions, owing to drastic departures from the idealistic assumptions for which the M-σ holds. Gravitational instabilities (Old et al. 2018) and member galaxy selection effects (Wojtak et al. 2018) are prime examples of complex systematics which violate M-σ assumptions and introduce error into dynamical cluster mass estimates. Considerable work has been done towards

Corresponding author: Matthew Ho
mho1@andrew.cmu.edu

quantifying and mitigating the uncertainties caused by these systematics (e.g. Wojtak et al. 2007; Mamon et al. 2013; Farahi et al. 2016, 2018; Abdullah et al. 2018). This proper modeling of cluster systems is crucial to the use of cluster abundance measurements for constraining cosmology.

Deep neural networks (DNNs; LeCun et al. 2015) are extremely versatile machine learning tools for modeling complex, non-linear relationships in data-rich environments such as cosmological analyses. In recent years, DNN modelling has met a large variety of useful applications, both broadly in physics (Carleo et al. 2019) and specifically in cosmology (e.g. Hoyle 2016; Lanusse et al. 2018; Ntampaka et al. 2019). In Ho et al. (2019), we showed that DNNs are able to mitigate systematics of dynamical cluster measurements to produce mass predictions with remarkably low bias and scatter. In addition, DNNs were computationally efficient to evaluate and robust to variations in sample richness, both requisite qualities for modern cluster mass estimators. In our comparative analysis, DNNs outperformed both simple and idealized M-σ analyses as well as other modern machine learning approaches (Ntampaka et al. 2015, 2016; Calderon & Berlind 2019).

While the increasingly precise inferences produced in (Ho et al. 2019) prove effective for the task of point mass inference, a natural extension would be to ask how one can quantify the uncertainty of our predictions. Estimates of measurement confidence are vital to recovering Bayesian constraints on cosmological parameters. Estimating Bayesian uncertainties of deep learning models has been an exceedingly active field of study in recent years (e.g. Neal 2012; Gal 2016; Caldeira & Nord 2020). While theoretically sound, the exact calculation of deep learning uncertainies is numerically intractable due to the necessary integration over hundreds of thousands of parameter posteriors. However, by assuming specific conjugate priors over neural network weights (e.g. Blundell et al. 2015; Gal & Ghahramani 2016), the computational complexity of this calculation can be drastically reduced. These approximate Bayesian uncertainties have been shown to accurately recover empirical variance in a wide variety of real datasets (e.g. Kendall & Gal 2017; Möller & de Boissière 2020), with particularly strong performance in modeling out-of-sample inputs (e.g. Gal & Ghahramani 2016).

In this paper, we seek to apply deep learning uncertainty estimation techniques to the cluster mass inference models presented in Ho et al. (2019). We discuss deep learning models in a Bayesian context and how assumptions of parameter priors can be used to tractably perform weight marginalization. Using a synthetic catalog of realistic cluster observations, we measure how well deep learning models can recover confidence intervals of dynamical cluster mass estimates. We investigate how choices of predictive distribution and parameter priors impact the quality of these deep learning predictions, both for individual clusters and for cosmological analyses. This paper is organized into the following sections: in Section 2, we describe the generation of the mock cluster catalog. In Section 3, we detail the theoretical considerations for Bayesian deep learning as well as the specific designs of the presented models. In Section 4, we evaluate model performance empirically and discuss the results. We summarize conclusions in Section 5. The code developed for this analysis is made publicly available on Github[1].

2. DATASET

In this section, we summarize important properties of the mock cluster observations used in this analysis. The mock catalog is a new realization of the contaminated mock observation procedure described in Ho et al. (2019). The catalog generation code is made available on Github[1] and pre-generated catalogs are available upon request.

The catalog is generated from a $z = 0.117$ snapshot of the MultiDark Planck 2 N-body simulation (MDPL2; Klypin et al. 2016), which assumes a ΛCDM cosmology consistent with 2013 *Planck* data (Planck Collaboration et al. 2014). Host halos and subhalos are identified in the MDPL2 simulation using the ROCKSTAR halo finder (MDPL2 Rockstar; Behroozi et al. 2013). We model clusters as host halos in the MDPL2 Rockstar catalog with spherical overdensity masses of $M_{200c} \geq 10^{13.5}\ h^{-1}\mathrm{M}_\odot$. Galaxies are painted onto subhalos via the UniverseMachine galaxy assignment procedure (Behroozi et al. 2019) and restricted to $M_{\mathrm{stellar}} \geq 10^{9.5}\ h^{-1}\mathrm{M}_\odot$. Clusters and galaxies in our sample inherit mass, position, and velocity from their respective halos in the MDPL2 Rockstar and UniverseMachine catalogs. Throughout the paper, we use the shorthand m to denote logarithmic spherical overdensity cluster masses,

$$m \equiv \log_{10}\left[M_{200\mathrm{c}}\ \left(h^{-1}\mathrm{M}_\odot\right)\right]. \tag{1}$$

The dynamical observables reported for each mock cluster are the line-of-sight velocities v_{los} and sky-projected radial positions R_{proj} of its selected member galaxies. For a given line-of-sight, v_{los} and R_{proj} are calculated for all galaxies in a large neighborhood around each simulated cluster from the perspective of a $z = 0$

[1] https://github.com/McWilliamsCenter/halo_cnn

observer. Member galaxies are then selected around each cluster in dynamical phase-space $\{v_{\mathrm{los}}, R_{\mathrm{proj}}\}$ via a large cylindrical selection cut. The selection cylinder is centered at each true cluster center and oriented along the line-of-sight, with half-length $v_{\mathrm{cut}} = 2500$ km s^{-1} and radius $R_{\mathrm{aperture}} = 1.6\ h^{-1}$Mpc. After the selection cut, valid mock clusters are further restricted to a richness cut of $N_{\mathrm{gal}} \geq 10$.

Mock cluster observations are taken from multiple lines-of-sight to augment the catalog and shape the mass distributions of the training, test, and validation sets. To mitigate biases introduced in model training, we construct the training set to have a constant number density of $dn/dm = 10^{-5.2}\ h^3\mathrm{Mpc}^{-3}\mathrm{dex}^{-1}$ across all cluster masses $M_{200\mathrm{c}} \geq\ 10^{13.5}\ h^{-1}\mathrm{M}_{\odot}$. To achieve this evenly-distributed training set, abundant low-mass clusters are downsampled and scarce high-mass clusters are upsampled. The upsampling procedure involves taking additional projections of the same clusters from various lines-of-sight. To avoid duplicate observations, these additional lines-of-sight are distributed with roughly even spacing on the unit sphere. To emulate realistic measurement conditions, the test set is weighted to follow the theoretical HMF of the MDPL2 simulation and is comprised of exactly three orthogonal line-of-sight projections per cluster. Lastly, a validation set is created by taking a disjoint 10% random sampling of the test set.

3. METHOD

In this section, we discuss the deep learning models and uncertainty estimation techniques used to reconstruct cluster masses from member galaxy dynamics. Due to the variety of possible treatments of this problem, we seek to implement several model designs and investigate how they perform in the context of cluster mass estimation. We present a suite of twelve models, each with a different combination of input type, predictive distribution, and weight priors.

3.1. *Input*

The models presented in this paper infer cluster masses from one of two member galaxy distributions: the univariate distribution of line-of-sight velocities, $\{v_{\mathrm{los}}\}$, or the joint distribution of line-of-sight velocities and projected radial distances, $\{v_{\mathrm{los}}, R_{\mathrm{proj}}\}$. We refer to these input types as one-dimensional (1D) or two-dimensional (2D) inputs, respectively. In Ho et al. (2019), we showed that the inclusion of R_{proj} information significantly improved the prediction performance of deep learning models. Here, we seek to investigate the impact of additional input dimensions on mass uncertainty estimation.

We use Kernel Density Estimators (KDE; Scott 2015, chap. 6) to preprocess each cluster's list of member galaxy observables (i.e. v_{los} and R_{proj}) into regular mappings of their distribution in dynamical phase space. KDEs are a non-parametric method for estimating a PDF given data. In essence, KDEs smooth the distribution of discrete data points into a continuous PDF using a fixed kernel function. This smoothing allows our model inputs to be more robust to fluctuations in sample richness, a desirable property for galaxy-based cluster observations. For both 1D and 2D input types, we use a Gaussian kernel with a fixed bandwidth scaling factor of $h_0 = 0.25$. Once estimated PDFs are constructed, they are sampled at regular intervals across the mock observation cylinder cut. 1D inputs are sampled at 48 evenly-spaced points along the range $|v_{\mathrm{los}}| \leq v_{\mathrm{cut}}$. 2D inputs are sampled on a regular grid of 48 × 48 points spanning the area defined by $|v_{\mathrm{los}}| \leq v_{\mathrm{cut}}$ and $0 \leq R_{\mathrm{proj}} \leq R_{\mathrm{aperture}}$, where v_{cut} and R_{aperture} are parameters chosen in mock catalog generation (§2). Example 1D and 2D inputs are shown in Figure 1 For more information on KDEs and their application to our preprocessing, refer to Ho et al. (2019).

3.2. *Deep Neural Networks*

Deep neural networks (DNNs; LeCun et al. 2015) are a class of parametric ML models which are commonly used for learning non-linear relationships in rich, complex datasets (e.g. Carleo et al. 2019). Within a DNN, input and output are related through a series of layered neural connections. Evaluation of a DNN involves passing input values through this sequence of neural layers, with each layer pass representing tensor multiplication with a weight matrix followed by an element-wise, non-linear activation function. DNNs can be viewed as a functional mapping $\mathbf{y} = f\left(\mathbf{x}; \boldsymbol{\theta}, \boldsymbol{\eta}\right)$ between inputs $\mathbf{x}$ and outputs $\mathbf{y}$, which is parameterized by weight matrices $\boldsymbol{\theta}$ and hyperparameters $\boldsymbol{\eta}$ (e.g. choices of neural architecture, activation function, etc.). In general and in this application, hyperparameters $\boldsymbol{\eta}$ are assumed to be fixed, though algorithms for optimizing these have been explored in recent literature (e.g. Zoph & Le 2016). For a more detailed explanation of DNNs and their evaluation, see Ho et al. (2019).

Classically, training a DNN involves attempting to find the optimal weight parameters $\boldsymbol{\theta}^*$ which produce the best mapping of inputs to outputs. The metric chosen to dictate model performance is called an objective loss function $\mathcal{L}$. Given a training set of example data $\mathcal{D} := \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^n$, we seek to minimize this loss func-

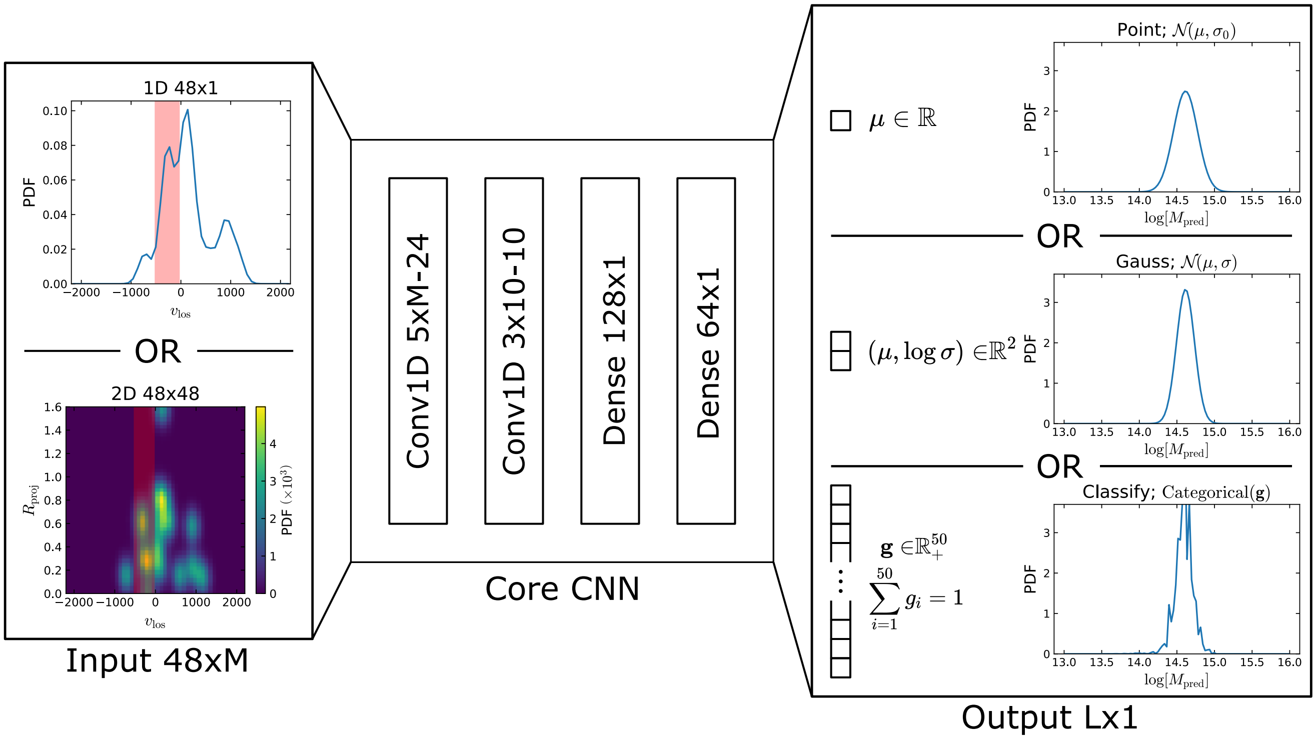

Figure 1. General CNN architecture used for our models. In our analysis, we explore a suite of twelve models, each with different choices of inputs, outputs, and weight priors. For all models, the central core CNN architecture is identical. Dropout connections are not shown here but are assumed to exist in between all layers for Dropout models. All layers utilize a rectified linear activation function (ReLU). In the diagram, convolutional layers are described using their filter shape and number of filters, respectively. Dense layers are characterized by their output layer shape. Here, we have used the notation $\mathbb{R}_+ := \{x | x \in \mathbb{R}, x > 0\}$.

tion over the space of possible parameters $\mathbf{\Theta}$.

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta} \in \mathbf{\Theta}} \sum_{i=1}^{n} \mathcal{L}\left(\mathbf{y}_i, f\left(\mathbf{x}_i; \boldsymbol{\theta}, \boldsymbol{\eta}\right)\right) \tag{2}$$

If we choose a convex loss function, supervised training of DNNs becomes a convex optimization problem, whose solution can be derived from simple gradient descent. Common choices of objective loss functions include mean squared error (for regression problems) and categorical cross-entropy (for classification problems). The power of neural networks arises from the fact that this optimization is numerically tractable, despite their highly non-linear structure and thousands to millions of free parameters.

3.3. *Bayesian Uncertainties*

As DNNs prove to be powerful and versatile tools for point regression and classification tasks, considerable work has gone into modelling their uncertainties (e.g. Gal 2016). Broadly, Bayesian uncertainties of deep learning models can be characterized as either aleatoric or epistemic. *Aleatoric uncertainties* capture intrinsic scatter in input-output relationships, wherein information encoded in input data is insufficient to precisely estimate true outputs, even given an ideal model. For example, the loss of 3D dynamical information inherent in projected cluster observations introduces aleatoric uncertainties. *Epistemic uncertainties* occur when training data or model flexibility is limited, such that we are unable to tightly constrain model parameters around the optimal setting, $\boldsymbol{\theta}^*$. In the context of deep learning cluster mass estimates, epistemic uncertainties would typically arise from insufficient network depth, training time, or training catalog diversity. Specific design choices and approximations must be made for proper, computationally-tractable modelling of these uncertainties. In this paper, we investigate several of these choices in the context of deep learning cluster mass estimates.

To capture aleatoric uncertainties, the functional output of a DNN can be used to dictate a distribution of outputs $p(\mathbf{y}|\mathbf{x}, \boldsymbol{\theta}, \boldsymbol{\eta})$ (e.g. Bishop 1994). For example, we can train a DNN to predict parameters of a univariate Gaussian. The final layer of the network would output estimates of means and variances, $f\left(\mathbf{x}; \boldsymbol{\theta}, \boldsymbol{\eta}\right) = (\mu, \log\sigma) \in \mathbb{R}^2$. This framework would allow neural networks to express not only what output predictions they can make, but also the statistical confidence that they have in those predictions. The type of predictive distribution is a design choice and should be closely representative of the true conditional distribution, $p(\mathbf{y}|\mathbf{x})$. Under ideal modelling conditions (i.e. infinite model flexibility, training data, and training time), aleatoric uncertainties are entirely sufficient for Bayesian modelling with DNNs.

However, under realistic modelling conditions, it is important to consider impacts of epistemic uncertainty on prediction. In traditional DNN training, we seek to find a single parameter setting $\hat{\boldsymbol{\theta}}$ which optimizes some loss metric $\mathcal{L}$ for the training data $\mathcal{D}$. However, even with an idealized training procedure, the recovered setting $\hat{\boldsymbol{\theta}}$ is often highly degenerate over the parameter space $\boldsymbol{\Theta}$. When training data is limited, it is possible to recover parameter settings which minimize loss over the training set but are not representative of the data at large. To model epistemic uncertainties, we marginalize predictive distributions over the conditional probability of all possible weight parameters given the training data.

$$p\left(\mathbf{y}|\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right)=\int p\left(\mathbf{y}|\mathbf{x},\boldsymbol{\theta},\boldsymbol{\eta}\right)p\left(\boldsymbol{\theta}|\boldsymbol{\eta},\mathcal{D}\right)d\boldsymbol{\theta}, \tag{3}$$

where $p\left(\mathbf{y}|\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right)$ is the weight-marginalized posterior distribution, $p\left(\mathbf{y}|\mathbf{x},\boldsymbol{\theta},\boldsymbol{\eta}\right)$ is the chosen predictive distribution, and $p\left(\boldsymbol{\theta}|\boldsymbol{\eta},\mathcal{D}\right)$ is the distribution of weight parameters informed by training data. The weight parameter distribution can be derived from Bayes rule,

$$p\left(\boldsymbol{\theta}|\boldsymbol{\eta},\mathcal{D}\right)\propto p\left(\mathcal{D}|\boldsymbol{\theta},\boldsymbol{\eta}\right)p\left(\boldsymbol{\theta}|\boldsymbol{\eta}\right), \tag{4}$$

where $p\left(\mathcal{D}|\boldsymbol{\theta},\boldsymbol{\eta}\right)=\prod_{i=1}^{n}p\left(\mathbf{y}_i|\mathbf{x}_i,\boldsymbol{\theta},\boldsymbol{\eta}\right)$ and $p\left(\boldsymbol{\theta}|\boldsymbol{\eta}\right)$ is a chosen weight prior. Eqn. 3 represents exact Bayesian inference, incorporating both aleatoric and epistemic uncertainties.

3.4. *Variational Inference*

Unfortunately, the full calculation of Eqn. 3 is numerically intractable for large DNNs. The integration over the space of hundreds of thousands of DNN weights is not feasible, even with highly efficient Monte Carlo methods. Variational inference is an alternative approach which instead interprets the posterior inference problem as an optimization. In this approach, we approximate the true weight distribution $p\left(\boldsymbol{\theta}|\boldsymbol{\eta},\mathcal{D}\right)$ with a variational distribution $q(\boldsymbol{\theta}|\hat{\boldsymbol{\phi}})$ whose form is chosen to simplify the integration in Eqn. 3. The optimal variational parameters $\hat{\boldsymbol{\phi}}$ can then be found by minimizing the metric distance (i.e. Kullback-Leibler divergence) between distributions $p\left(\boldsymbol{\theta}|\boldsymbol{\eta},\mathcal{D}\right)$ and $q(\boldsymbol{\theta}|\hat{\boldsymbol{\phi}})$. This minimization objective, referred to as $\mathcal{F}(\mathcal{D},\boldsymbol{\phi})$, is often called the variational free energy or the expected lower bound (ELBO).

$$\begin{aligned}\mathcal{F}(\mathcal{D},\boldsymbol{\phi})&=\mathrm{KL}\left[q\left(\boldsymbol{\theta}|\boldsymbol{\phi}\right)||p\left(\boldsymbol{\theta}|\boldsymbol{\eta},\mathcal{D}\right)\right]\\&=\mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\phi})}\left[p\left(\mathcal{D}|\boldsymbol{\theta},\boldsymbol{\eta}\right)\right]+\mathrm{KL}\left[q\left(\boldsymbol{\theta}|\boldsymbol{\phi}\right)||p\left(\boldsymbol{\theta}|\boldsymbol{\eta}\right)\right]\end{aligned} \tag{5}$$

where $\mathbb{E}_{q(\boldsymbol{\theta}|\boldsymbol{\phi})}[\cdot]$ represents the expectation over $q\left(\boldsymbol{\theta}|\boldsymbol{\phi}\right)$. Equipped with the analytic forms of $q\left(\boldsymbol{\theta}|\boldsymbol{\phi}\right)$, $p\left(\mathbf{y}|\mathbf{x},\boldsymbol{\theta},\boldsymbol{\eta}\right)$, and $p\left(\boldsymbol{\theta}|\boldsymbol{\eta}\right)$, we can minimize the objective loss in Eqn. 5 over the space of $\boldsymbol{\phi}$'s using optimization techniques such as gradient descent. Under this variational technique, Bayesian posterior inference then reduces to a two step process: a training stage wherein the optimal variational parameters $\hat{\boldsymbol{\phi}}$ are determined from data and an inference stage which folds $q(\boldsymbol{\theta}|\hat{\boldsymbol{\phi}})$ into Eqn. 3.

The functional forms of variational distributions $q\left(\boldsymbol{\theta}|\boldsymbol{\phi}\right)$ and priors $p\left(\boldsymbol{\theta}|\boldsymbol{\eta}\right)$ are design choices. Several forms of variational distributions have been implemented in the literature (e.g. Gal & Ghahramani 2016; Blundell et al. 2015), but there lacks a consensus for an ideal choice. The most trivial variational distribution is a delta function with $q\left(\boldsymbol{\theta}|\boldsymbol{\phi}\right)=\delta\left(\boldsymbol{\theta}-\boldsymbol{\phi}\right)$. Here, we assume epistemic uncertainty to be negligible, as Eqn. 3 reduces to the chosen predictive distribution with $\boldsymbol{\theta}=\hat{\boldsymbol{\phi}}$. If we set the predictive distribution to be a fixed-mean Gaussian or a multinomial, the objective loss simplifies to the classical mean squared error or categorical cross-entropy, respectively. Furthermore, the inclusion of Gaussian or Laplacian priors $p\left(\boldsymbol{\theta}|\boldsymbol{\eta}\right)$ respectively adds L2 or L1 regularization penalties to weight parameters in the loss function.

Another common choice of weight prior is a multivariate Bernoulli distribution. Gal & Ghahramani (2016) investigated the nature of a Bernoulli-distributed $q\left(\boldsymbol{\theta}|\boldsymbol{\phi}\right)$ with a zero-mean, diagonal Gaussian $p\left(\boldsymbol{\theta}|\boldsymbol{\eta}\right)$. In their implementation, they utilized the popular regularization technique, Dropout, to perform stochastic integration (Eqn. 3). In both the training and inference stages, Dropout layers are allowed to randomly set some fraction, $p_d\in[0,1]$, of the weight parameters equal to 0. The Dropout layers are stochastic, causing each functional evaluation of the model to use a different weight configuration. During training, this acts to regularize the iterative updates of stochastic gradient descent (Srivastava et al. 2014). During inference, one can average many realizations of the Dropout layers to effectively produce a Monte Carlo estimate of the model output. Gal & Ghahramani (2016) showed that such a training and evaluation procedure approximates a Gaussian Process and is able to accurately recover uncertainties for both in- and out-of-sample data.

Table 1. Configuration of Investigated Models

Model Name	$\mathbf{x}$	$f(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\eta})$	$p(m\vert\mathbf{x},\boldsymbol{\theta},\boldsymbol{\eta})$	$q(\boldsymbol{\theta}\vert\boldsymbol{\phi})$
1DPoint	$\{v_{\rm los}\}$	$(\mu)\in\mathbb{R}$	$\mathcal{N}(m;\mu,\sigma^2_{\mathcal{D}})$	$\prod_i \delta(\theta_i-\phi_i)$
1DPoint-d	$\{v_{\rm los}\}$	$(\mu)\in\mathbb{R}$	$\mathcal{N}(m;\mu,\sigma^2_{\mathcal{D}})$	$\prod_i$ Bernoulli $[\delta(\theta_i-\phi_i);p_d]$
1DGauss	$\{v_{\rm los}\}$	$(\mu,\log\sigma)\in\mathbb{R}^2$	$\mathcal{N}(m;\mu,\sigma^2)$	$\prod_i \delta(\theta_i-\phi_i)$
1DGauss-d	$\{v_{\rm los}\}$	$(\mu,\log\sigma)\in\mathbb{R}^2$	$\mathcal{N}(m;\mu,\sigma^2)$	$\prod_i$ Bernoulli $[\delta(\theta_i-\phi_i);p_d]$
1DClass	$\{v_{\rm los}\}$	$\mathbf{g}\in\mathbb{R}^{50}$	Categorical $[m;S(\mathbf{g})]$	$\prod_i \delta(\theta_i-\phi_i)$
1DClass-d	$\{v_{\rm los}\}$	$\mathbf{g}\in\mathbb{R}^{50}$	Categorical $[m;S(\mathbf{g})]$	$\prod_i$ Bernoulli $[\delta(\theta_i-\phi_i);p_d]$
2DPoint	$\{R_{\rm proj},v_{\rm los}\}$	$(\mu)\in\mathbb{R}$	$\mathcal{N}(m;\mu,\sigma^2_{\mathcal{D}})$	$\prod_i \delta(\theta_i-\phi_i)$
2DPoint-d	$\{R_{\rm proj},v_{\rm los}\}$	$(\mu)\in\mathbb{R}$	$\mathcal{N}(m;\mu,\sigma^2_{\mathcal{D}})$	$\prod_i$ Bernoulli $[\delta(\theta_i-\phi_i);p_d]$
2DGauss	$\{R_{\rm proj},v_{\rm los}\}$	$(\mu,\log\sigma)\in\mathbb{R}^2$	$\mathcal{N}(m;\mu,\sigma^2)$	$\prod_i \delta(\theta_i-\phi_i)$
2DGauss-d	$\{R_{\rm proj},v_{\rm los}\}$	$(\mu,\log\sigma)\in\mathbb{R}^2$	$\mathcal{N}(m;\mu,\sigma^2)$	$\prod_i$ Bernoulli $[\delta(\theta_i-\phi_i);p_d]$
2DClass	$\{R_{\rm proj},v_{\rm los}\}$	$\mathbf{g}\in\mathbb{R}^{50}$	Categorical $[m;S(\mathbf{g})]$	$\prod_i \delta(\theta_i-\phi_i)$
2DClass-d	$\{R_{\rm proj},v_{\rm los}\}$	$\mathbf{g}\in\mathbb{R}^{50}$	Categorical $[m;S(\mathbf{g})]$	$\prod_i$ Bernoulli $[\delta(\theta_i-\phi_i);p_d]$

Note—Models are presented with the design choices made for their inputs $\mathbf{x}$, functional outputs $f(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\eta})$, predictive distributions $p(m|\mathbf{x},\boldsymbol{\theta},\boldsymbol{\eta})$, and variational weight distribution $q(\boldsymbol{\theta}|\boldsymbol{\phi})$. For Point models, $\sigma^2_{\mathcal{D}}$ is equal to the mean squared error of model predictions after training. For clarity, the dependence of functional outputs μ,σ, and $\mathbf{g}$ on $(\mathbf{x};\boldsymbol{\theta},\boldsymbol{\eta})$ has been suppressed. We use the notation $D[x;p_1,p_2,\dots]$ to denote an evaluation of the PDF of distribution D with parameters $(p_1,p_2,\dots)$ at x. $S(\cdot)$ denotes the softmax function.

3.5. *Models*

The models presented in this paper attempt to infer logarithmic cluster mass, m (Eqn. 1), from mappings of dynamical phase space, $\mathbf{x}$ (§3.1). All models are set up with a fixed neural architecture, $\boldsymbol{\eta}$, and trained with a labeled set of mock cluster observations, $\mathcal{D}:=\{(\mathbf{x}_i,m_i)\}_{i=1}^n$. Using the approximate Bayesian inference techniques described in §3.3, each model outputs a posterior distribution over logarithmic cluster masses, $p(m|\mathbf{x},\boldsymbol{\eta},\mathcal{D})$.

We investigate the impact of various design choices on the performance of our models. We implement a suite of twelve models, each with a different configuration of input type $\mathbf{x}$, predictive distribution $p(m|\mathbf{x},\boldsymbol{\theta},\boldsymbol{\eta})$, and variational distribution $q(\boldsymbol{\theta}|\boldsymbol{\phi})$. For modelling aleatoric uncertainty, we choose one of three predictive distributions: a fixed-width Gaussian (Point), a variable-width Gaussian (Gauss), and a 50-bin Categorical distribution spanning $13\le m\le 16$ (Class). For modelling epistemic uncertainty, we implement two forms of variational distributions: a standard Dirac delta function and a Bernoulli distribution (Gal & Ghahramani 2016). Models implementing Dropout marginalization with a Bernoulli variational distribution are named with the suffix '-d'. Table 1 contains a list of each model and its respective configuration. A schematic of each model's architecture is shown in Figure 1. The posterior distributions and objective loss functions derived for each model are tabulated in Tables 3, respectively.

Hyperparameters $\boldsymbol{\eta}$ and weight priors $p(\boldsymbol{\theta}|\boldsymbol{\eta})$ are held constant for all models. The core architecture of each model is a Convolutional Neural Network (CNN; LeCun et al. 1998). CNNs are widely used in computer vision for their ability to learn patterns in localized subregions of input data. Here, we use CNNs to identify artifacts in regions of dynamical phase space such as groups of interloping galaxies or cluster mergers. The specific CNN architecture applied here was introduced in Ho et al. (2019) and is shown in Figure 1. For each model, we use a zero-mean, diagonal Gaussian prior on model weights $p(\boldsymbol{\theta}|\boldsymbol{\eta})=\mathcal{N}(0,\lambda\mathbb{I})$ where $\lambda=10^{-4}$. Inclusion of this prior amounts to adding a weight decay regularization term $\lambda||\boldsymbol{\theta}||_2^2$ to each objective loss function.

3.6. *Implementation*

For models using a delta function variational distribution, training and inference are exactly equivalent to classical DNN models. Since this distribution assumes $\boldsymbol{\theta}=\boldsymbol{\phi}$, optimization reduces to solving Eqn. 2 via gradient descent for the loss functions shown in Table 3. Inference simplifies to an evaluation of our chosen predictive distribution at the optimized parameterization, $p(m|\mathbf{x},\boldsymbol{\eta},\mathcal{D})=p\big(m|\mathbf{x},\hat{\boldsymbol{\theta}},\boldsymbol{\eta}\big)$.

We follow the procedure detailed in Gal & Ghahramani (2015) to implement weight marginalization for models using Bernoulli variational distributions. During both model training and inference, we include Dropout layers after all existing neural layers in the core network architecture (Figure 1). Dropout layers do no tensor operations, but instead randomly set some prescribed fraction of values from their input tensor equal to 0. This effectively makes the functional output of our neural network stochastic, as each pass includes random realizations of the several Dropout layers. Gal & Ghahramani (2016) showed that using gradient descent to minimize the loss functions in Table 3 under these stochastic evaluation conditions solves Eqn. 5. To perform inference, we approximate marginalization over the variational distribution by combining the network outputs of many realizations of the Dropout layers (Table 3). In our implementation, we set our dropout rate to $p_d = 0.1$ and take $T = 100$ realizations of the neural evaluation to produce inference.

We use a 10-fold cross-validation scheme to train and evaluate our models. For a given fold, we train on 9/10 of the cluster candidates in our catalog and test on the remaining, independent 1/10. This process cycles for 10 folds until predictions have been made for the entire test set. Cluster candidates are grouped along with their rotated LOS duplicates in the training-test split, such that we are never training and testing on the same cluster from different LOSs. This ensures independence of training and testing data for each fold. On average, there are $\sim 10,000$ training and $\sim 7,000$ test cluster candidates for a given fold.

During training, we use the Adam optimization procedure (Kingma & Ba 2014) with a learning rate of 10^{-3} and a batch size of 100. We achieve loss convergence within 40 epochs of training. All models are implemented using the *Keras*[2] deep learning library with a *Theano*[3] backend.

4. RESULTS

We quantify the validity of our uncertainty estimation techniques in the context of astronomical and cosmological analyses. The objectives of our analysis are threefold. First, we confirm that these models accurately reproduce the point prediction performance presented in Ho et al. (2019). Second, we characterize the nature of our uncertainty predictions, including how well our predictive distributions match the empirical distribution of cluster masses. All analyses are conducted on the contaminated cluster catalog described in §2 wherein true masses are known. Model predictions are made using the ten-fold training and inference procedure described in §3.5.

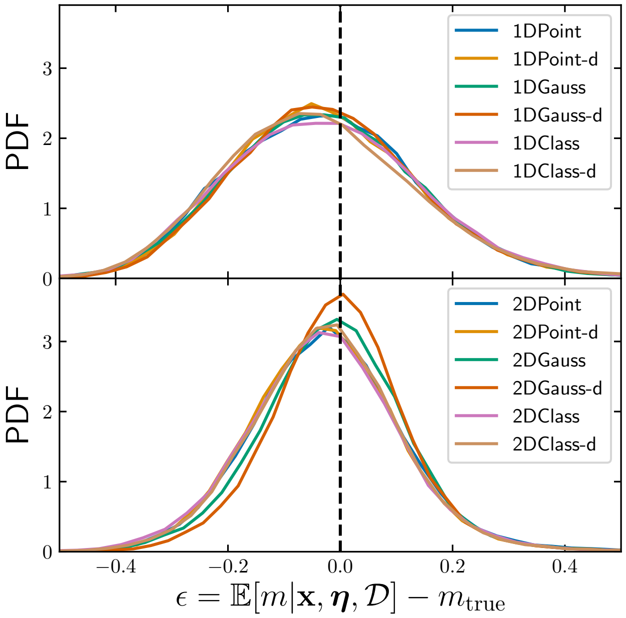

Figure 2. Distribution of point prediction residuals (Eqn. 7) for models in Table 1. Point residual distributions are averaged over all cross-validation folds of test clusters in the mass range $14 \leq m_{\rm true} \leq 15$.

4.1. *Point Predictions*

We evaluate the accuracy and Gaussianity of point predictions made by our models. In this context, we define point predictions to be the mean of the estimated posterior distribution for logarithmic cluster mass (Eqn. 1),

$$\mathbb{E}\left[m|\mathbf{x}, \boldsymbol{\eta}, \mathcal{D}\right] := \int m \; p(m|\mathbf{x}, \boldsymbol{\eta}, \mathcal{D}) dm. \qquad (6)$$

Following from this definition, we utilize the following characterization of the point residual ϵ as the difference between the point prediction and true logarithmic mass,

$$\epsilon := \mathbb{E}\left[m|\mathbf{x}, \boldsymbol{\eta}, \mathcal{D}\right] - m_{\rm true}. \qquad (7)$$

It is self-evident that, for this choice of point prediction, the models utilizing constant-variance Gaussian predictive distributions, 1DPoint and 2DPoint, are functionally equivalent to the models presented in Ho et al. (2019) and should have equivalent performance. We also note that other choices of cumulative statistics such as the median or mode of the predictive distribution are also valid point predictors of cluster mass, though they are not considered here.

Our analysis shows that point residuals produced by each model in our suite have low scatter, demonstrate

[2] https://keras.io/

[3] http://deeplearning.net/software/theano/

Table 2. Descriptive Statistics of Model Performance.

Model Name	$\tilde{\epsilon} \pm \Delta\epsilon^{a}$	$\sigma_\epsilon{}^{b}$	γ^{b}	κ^{b}	$\widetilde{\mathrm{Var}}\left[m\vert\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right]^{c}$	$\hat{r}(0.5)^{d}$	16-84 EPR^{d}	5-95 EPR^{d}
1DPoint	$-0.032^{+0.168}_{-0.167}$	0.172	0.427	0.758	0.032	0.425	0.697	0.919
1DPoint-d	$-0.036^{+0.170}_{-0.156}$	0.170	0.469	0.911	0.035	0.415	0.725	0.931
1DGauss	$-0.033^{+0.173}_{-0.162}$	0.173	0.382	0.774	0.026	0.425	0.682	0.891
1DGauss-d	$-0.030^{+0.167}_{-0.157}$	0.169	0.497	1.033	0.036	0.427	0.773	0.945
1DClass	$-0.034^{+0.178}_{-0.169}$	0.178	0.429	0.873	0.030	0.463	0.673	0.904
1DClass-d	$-0.045^{+0.181}_{-0.160}$	0.178	0.581	1.203	0.033	0.430	0.715	0.929
2DPoint	$-0.024^{+0.129}_{-0.129}$	0.138	0.362	1.385	0.024	0.422	0.752	0.935
2DPoint-d	$-0.030^{+0.127}_{-0.126}$	0.134	0.353	1.372	0.027	0.406	0.776	0.949
2DGauss	$-0.011^{+0.119}_{-0.125}$	0.132	0.193	1.535	0.018	0.460	0.708	0.915
2DGauss-d	$-0.003^{+0.110}_{-0.113}$	0.123	0.333	1.886	0.023	0.488	0.778	0.947
2DClass	$-0.030^{+0.128}_{-0.131}$	0.140	0.289	1.762	0.020	0.433	0.680	0.904
2DClass-d	$-0.026^{+0.125}_{-0.127}$	0.136	0.340	2.015	0.021	0.446	0.711	0.925

[a] Point residual median and 16-84 percentile range (dex)

[b] Point residual standard deviation scatter (dex), skewness, and excess kurtosis, respectively

[c] Average posterior variance

[d] Empirical percentile (Eqn. 10) median, 16-84 range, and 5-95 range, respectively. R_1-R_2 empirical percentile ranges are equivalent to $\hat{r}(R_2\%) - \hat{r}(R_1\%)$.

Note—Quantities are averaged over all cross-validation folds of test clusters in the mass range $14 \leq m_{\mathrm{true}} \leq 15$.

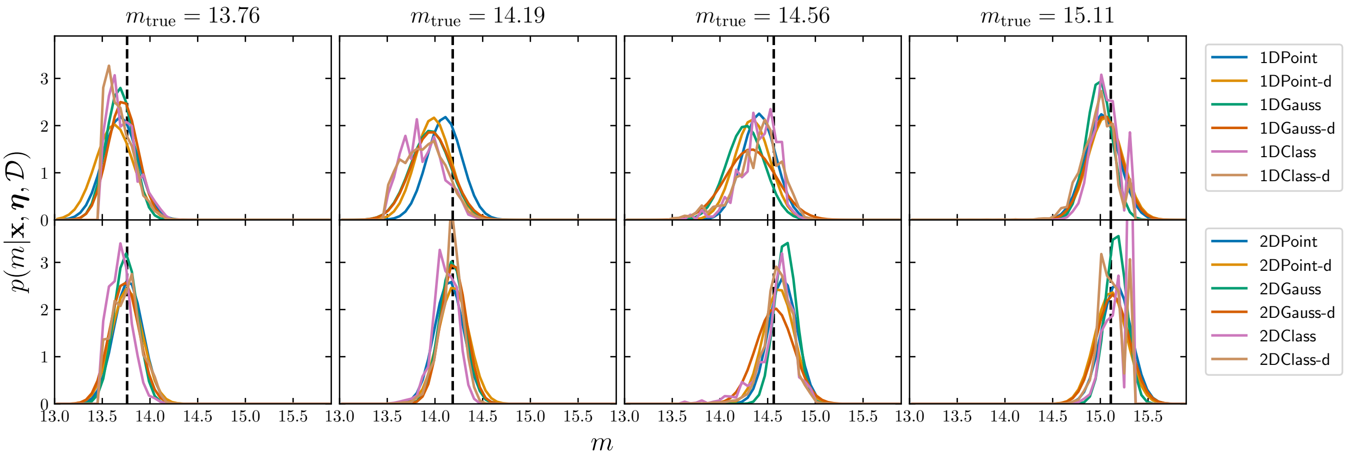

Figure 3. Posterior distributions estimated by each model in Table 1 for four randomly-selected clusters across a variety of true masses. Each column shows mass posteriors generated from a single line-of-sight projection of a mock cluster in our test set. Each cluster's true logarithmic mass, m_{true}, is stated in the column title and plotted as a black dashed line. For clarity, 1D and 2D model distributions are shown on separate rows.

very low statistical bias, and are roughly Gaussian-distributed when averaged over the test data set. This is shown in Figure 2 and Table 2 where we have calculated the empirical distributions of point residuals and their cumulative statistics. The scatter of point estimate residuals for 1D and 2D models are approximately equal to those described in Ho et al. (2019), where 1D and 2D scatters were recorded to be 0.174 dex and 0.132 dex, respectively. The difference in predictive scatter between 1D and 2D models is motivated by the inclusion of supplemental R_{proj} information in 2D inputs. In addition, measurements of skewness γ and excess kurtosis κ of the residual distributions are consistent with near-Gaussianity.

The results in this section are consistent with the findings of Ho et al. (2019) and indicate that the extension of deep learning mass estimation models to more complex predictive distributions with higher dimensional outputs does not diminish their point prediction performance. The deep learning machinery within each model in our suite is able to capture the same information from the input for addressing the simple task of point prediction.

4.2. *Uncertainty Estimation*

Figure 3 shows posteriors produced by all investigated models for four randomly-selected mock clusters. In the examples shown, all models are able to accurately recover true cluster masses to within a 90% confidence interval. Each model assigns probability to small, localized regions of logarithmic masses roughly centered at m_{true}. Classification models only assign probability to mass bins where there exists training data (i.e. $13.5 \leq m \leq 15.3$). For a given input type, model posteriors are strongly consistent. Classification models, whose posterior family is highly flexible, produce posteriors which are near-Gaussian, lending to the fact that assumptions of Gaussian predictive distributions for Point and Gauss models are well-founded.

To compare recovered uncertainties, we approximate each model posterior as a point estimate with Gaussian noise of variance $\text{Var}\left[m|\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right]$. Here, we define posterior variance as:

$$\text{Var}\left[m|\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right] := \mathbb{E}\left[m^2|\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right] - \mathbb{E}\left[m|\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right]^2 . \quad (8)$$

The distribution of posterior variances across our test set is shown in Figure 4.

We observe that estimated posterior variances are non-constant for all models except 1DPoint and 2DPoint, whose variances are fixed by construction. For 1DPoint-d and 2DPoint-d models, posterior variances are largely independent of true mass. Scatter in these variance estimates arises entirely from the stochastic estimates of epistemic uncertainty. For Gauss and Class models, the estimated posterior variance exhibits a noticeable dependence on true mass. For these models, posterior variance is low for clusters at the edges of our test mass range and high for clusters around $m_{\text{true}} \sim 14$. This dependence is contrary to expectations of galaxy-based cluster mass estimators, where scatter is expected to decrease with increasing cluster richness and mass (Wojtak et al. 2018). However, the mass-dependence of our models' posterior variances is likely biased by the mass cut placed on our training set (i.e. $m_{\text{true}} \geq 13.5$). Because of this mass cut, models are trained to minimize any probability assigned to mass estimates lower than $m = 13.5$. This cut thereby removes a considerable amount of variability in low mass cluster predictions. This same reasoning applies to posterior variances of high mass clusters, and its reduction effects can be observed in Figure 4. However, in the safe inner range of cluster masses ($14 \leq m_{\text{true}} \leq 15$), posterior variance decreases with increasing cluster mass, as expected. To mitigate the impact of the mass cut biases, future work could explore solutions such as lowering the training mass cut or reweighting low mass posteriors.

Design choices such as input type and variational distribution directly impact the magnitude of recovered posterior variances (Table 2). On average, the posterior variance estimated by 2D models is 69% that of 1D models. This is expected, as the additional R_{proj} information given to 2D models allows recovery of tighter constraints on cluster mass (Ho et al. 2019). Alternatively, the use of a Bernoulli variational weight distribution over a delta function increases posterior variance by 17% on average. This difference amounts to inclusion of epistemic uncertainties, which are assumed to be negligible when using a delta function.

To validate posterior recovery, we compare the posterior distributions predicted by our investigated models to the empirical distribution of true masses in the test set. To do so, we compare predictive percentiles r recovered by our model posteriors to the corresponding empirical percentiles $\hat{r}(r)$ present in the data. We first define the predictive quantile $m_q\left(r;\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right)$ as the logarithmic mass which satisfies:

$$r = \int_{-\infty}^{m_q(r;\mathbf{x},\boldsymbol{\eta},\mathcal{D})} p\left(m|\mathbf{x},\boldsymbol{\eta},\mathcal{D}\right)\, dm, \quad (9)$$

for a percentile r and posterior $p(m|\mathbf{x},\boldsymbol{\theta},\boldsymbol{\eta})$. We then define empirical percentile $\hat{r}(r)$ as the fraction of clusters in our test set $\mathcal{D}_{\text{test}} = \{(\mathbf{x}_i, m_i)\}_{i=1}^{N_{\text{test}}}$ with masses less than or equal to the predictive quantile.

$$\hat{r}(r) = \frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \mathbb{I}\left[m_i \leq m_q\left(r;\mathbf{x}_i,\boldsymbol{\eta},\mathcal{D}\right)\right], \quad (10)$$

where $\mathbb{I}[\cdot]$ is the indicator function. Under this construction, a perfectly calibrated posterior would produce predictive percentiles which exactly match empirical percentiles, $\hat{r}(r) = r$ for $r \in [0,1]$. A model posterior which consistently biases towards low masses would produce $\hat{r}(r) \leq r$ for $r \in [0,1]$, and vice versa for high mass biasing. If a model posterior is unbiased but underestimates variance, then $\hat{r}(r) \geq r$ for $r \in [0, 0.5]$ and $\hat{r}(r) \leq r$ for $r \in [0.5, 1]$, and vice versa for overestimation. This metric allows us to compare, on average, how well our models recover percentiles and confidence intervals of the true cluster mass distribution. A similar technique

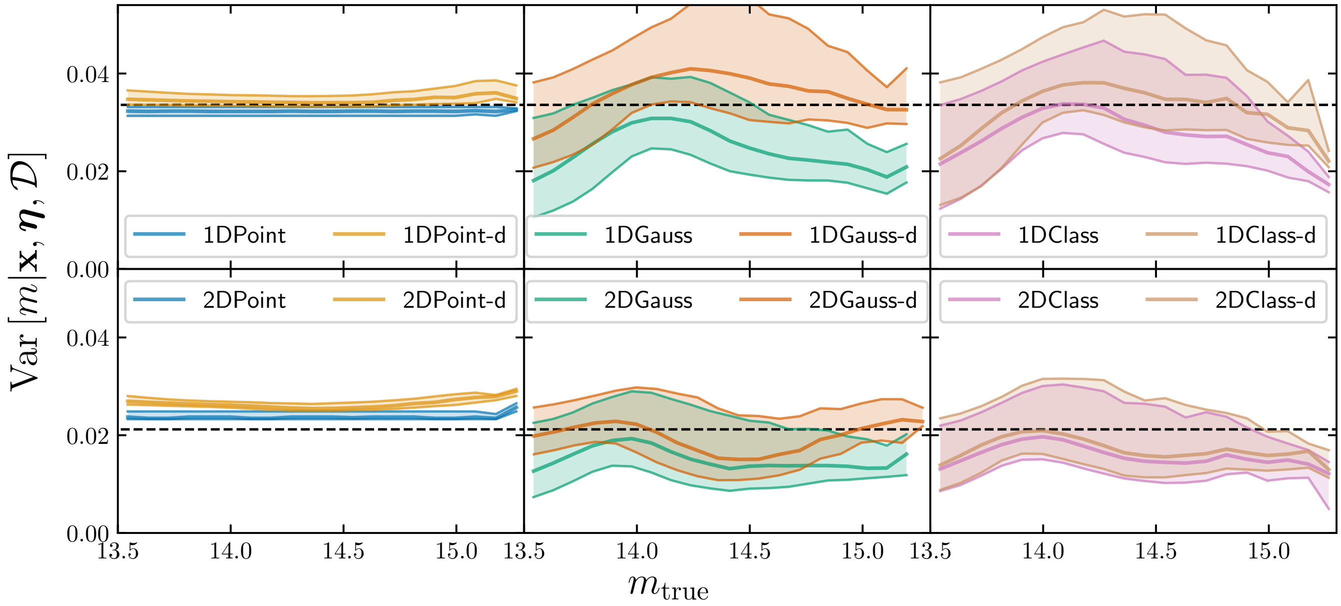

Figure 4. Posterior variance of mock cluster mass estimates (Eqn. 8) as a function of true logarithmic mass. Standard deviation distributions are binned along true mass and shown at their median and 16-84 percentile range. Binned distributions are averaged over all cross-validation folds in the test set. For each input type, we plot a black dashed line representing the ~~of~~ point residual variances σ_ϵ^2, as reported by Table 2. We note that, although posterior variances of 1DPoint and 2DPoint models are fixed by construction, we observe small variations in their estimates on account of the cross-validation training and evaluation procedure.

was used to validate posterior quantiles in Cook et al. (2006).

Figure 5 shows empirical percentiles for all investigated models. To demonstrate mass-dependent biases, we show separate lines for empirical percentiles calculated from low ($m_{\rm true} < 14$), medium ($14 \leq m_{\rm true} < 15$), and high ($15 \leq m_{\rm true}$) clusters. We observe noticeable mean reversion for model predictions on the edges of our training set. All model posteriors tend to bias towards the middle of our mass range, meaning that clusters with high true masses are assigned lower mass posteriors and vice versa. This mean reversion is an inherent artefact of the interpolating behavior of machine learning models. As non-analytic models, the DNNs implemented in this analysis struggle to extrapolate predictions to the edges of our dataset, but perform well in the inner regions. This systematic bias was also observed for point estimate masses in Ntampaka et al. (2016) and Ho et al. (2019).

For observed clusters in the reliable inner mass range ($14 \leq m_{\rm true} < 15$), predictive percentiles closely resemble empirical percentiles with a slight bias towards low mass predictions. We characterize this bias by the median empirical percentile $\hat{r}(0.5)$, as tabulated in Table 2. We find that median empirical percentiles are less than 50% by at least 1.2% (2DGauss-d) and at most 9.4% (2DPoint-d). As a result, model predictions of median cluster mass can be expected to fall, on average, between the 40th and 49th percentile of the true distribution, $p(m|\mathbf{x})$. This slight negative bias echos the findings of the point prediction analysis (§4.1) in Figure 2 and Table 2.

We construct a metric to quantify our models' calibration of predictive confidence intervals. We define the R_1-R_2 empirical percentile range (EPR) as the fraction of true masses captured between the R_1 and R_2-th quantiles of our predictive posteriors. This quantity is equivalent to $\hat{r}(R_2\%) - \hat{r}(R_1\%)$ and should equal $R_2\% - R_1\%$ for an ideal model. Table 2 tabulates the empirical percentile ranges for 16-84 and 5-95 confidence intervals. Despite median biases, model posteriors are able to recover empirical confidence intervals with a high degree of accuracy. All models are able to recover both 16-84 and 5-95 confidence intervals to within ±10% of their empirical value, with 2/3 of models estimating confidence intervals to within ±5%. The best performing models, 1DGauss and 2DClass, are able to reproduce both 16-84 and 5-95 confidence intervals to within 1% of their empirical range. The 16-84 and 5-95 EPRs of a majority of models (2/3) tend to be greater than their fiducial values, suggesting that these models are slightly overpredicting predictive variance.

By a small margin, Point models report the worst recovery of empirical percentiles among the various choices

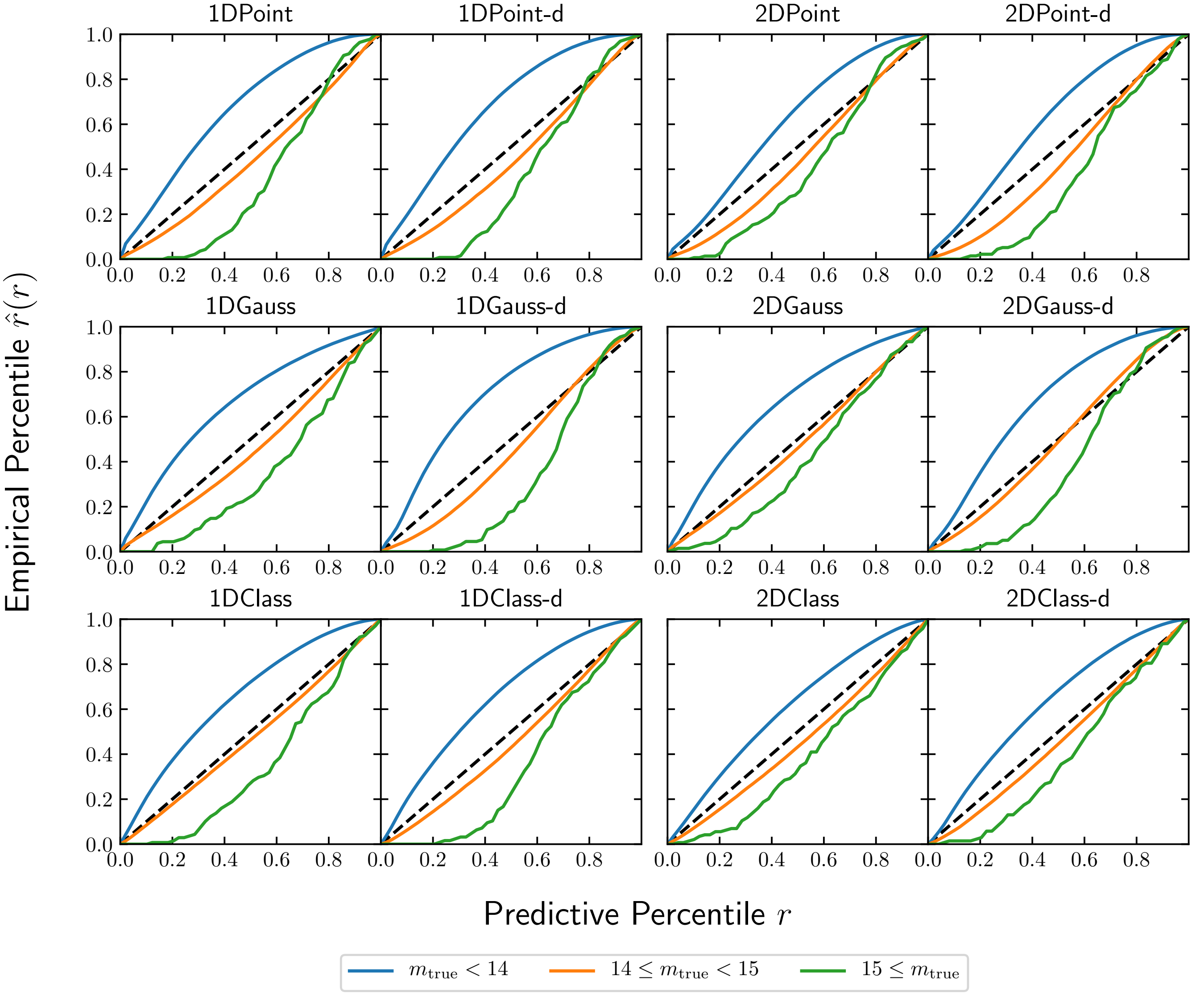

Figure 5. Empirical percentiles $\hat{r}(r)$ (Eqn. 10) as a function of predictive percentile r for all investigated models (Table 1). As implemented here, empirical percentiles capture the fraction of times the true mass of a cluster sample in our test set falls below the r-th quantile of our model posterior (Eqn. 9). We show empirical percentiles recovered for test clusters in three disjoint mass ranges.

of predictive distributions. Apart from the 16-84 EPR calculated for 1DPoint, performance metrics for Point models appear to deviate the most from fiducial values. The performances of Gauss and Class models appear to be roughly equal, with both model classes reporting the best recovery of empirical percentile ranges (i.e. 1DGauss and 2DClass, respectively). This suggests that the added variance parameter of Gauss posteriors is well-utilized in our cluster mass estimation task. However, further flexibility (e.g. non-Gaussian posteriors in Class models) does not necessarily improve our predictive performance.

For the metrics reported in Table 2, there is little to no model performance improvement from the inclusion of a Bernoulli variational distribution. In all cases, models with Bernoulli-distributed weight priors have larger and further deviated 16-84 and 5-95 EPRs than their delta function weight prior counterparts. The inclusion of Bernoulli weight priors seems to consistently overestimate predictive uncertainties. This suggests that our training data set is sufficiently large to tightly constrain weight parameters and that epistemic uncertainties can be safely assumed to be negligible. Other applications of Bernoulli-distributed weighting have found that improvements are very model dependent and should be tested empirically before practical application (Gal & Ghahramani 2015; Caldeira & Nord 2020).

5. CONCLUSION

This paper is an extension of Ho et al. (2019) in which we implement modern Bayesian uncertainty reconstruction techniques for deep learning mass estimates of galaxy clusters. The deep learning models learn logarithmic cluster mass $m = \log_{10}\left[M_{200c}\ \left(h^{-1}\mathrm{M}_\odot\right)\right]$ from dynamical cluster observables such as line-of-sight velocities (v_{los}) and projected radial distances (R_{proj}) of member galaxies. We seek to estimate posterior distributions $p(m|\mathbf{x}, \boldsymbol{\eta}, \mathcal{D})$ over cluster masses given dynamical inputs $\mathbf{x}$, network architectures $\boldsymbol{\eta}$, and training data $\mathcal{D}$. We review methods for deep learning uncertainty estimation and investigate several configurations of model design choices in our implementation. The full list of models and their respective designs is given in Table 1.

We train and evaluate our models using a mock cluster observation catalog derived from a single redshift snapshot of a dark matter simulation. The mock catalog is designed to incorporate physical and selection systematics which impact real dynamical observations of galaxy clusters. We use a 10-fold cross-validation scheme to train and test our models. We measure performance metrics which characterize how well each model can both predict point estimates of cluster mass as well as recover full mass posteriors. The findings of our analysis are as follows:

- To enable reconstruction of mass posteriors, we introduce additional complexity to the models first presented in Ho et al. (2019). We find that this additional complexity does not diminish our ability to estimate point masses efficiently and precisely. All model implementations produce point mass estimates with Gaussian scatter at the same level as that reported in Ho et al. (2019).

- Mass posteriors from all models in our suite are mutually consistent and assign probability to a small, localized region of cluster masses centered at the true cluster mass. The highly-flexible posteriors of Class models converge to a near-Gaussian shape, suggesting that model assumptions of a Gaussian predictive distribution are well-founded.

- Inclusion of R_{proj} information in model inputs reduces predictive variance by 31% on average. Modelling epistemic uncertainties with the Dropout approximation (Gal & Ghahramani 2016) increases predictive variance by 17% on average.

- Model predictions at the edges of our test set exhibit a noticeable mean-reversion effect, biasing mass posteriors towards the center of our mass range. In the inner region of our test set, model posteriors are slightly biased towards low masses on average.

- All models are able to recover both 16-84 and 5-95 confidence intervals to within ±10% of their empirical value. The best performing models are able to recover 16-84 and 5-95 confidence intervals on cluster mass to within 1% of their empirical value.

- Modelling of epistemic uncertainties does not improve posterior recovery of our models. The impacts of epistemic uncertainties are negligible relative to posterior variances introduced by aleatoric uncertainties. This suggests that our mock catalog and training procedure are sufficient to fit the mass-observable relation.

We note that the results presented here are only tested for the simplistic mock catalogs described in §2 and may not necessarily hold in the presence of other realistic observational systematics such as complex survey selection functions and photon collisions. We also remark that the approximate Bayesian technique described here is not the only method for reconstructing uncertainties from DNNs. An alternative method introduced by Kodi Ramanah et al. (2020) utilizes neural flows to infer prediction uncertainties and achieves promising results. In addition, they apply their method on spectroscopic data from the the NASA/IPAC Extragalactic Database to make preliminary dynamical mass estimates of several real galaxy clusters.

In conclusion, we design and investigate a numerical procedure for performing approximate Bayesian inference on DNNs for galaxy cluster mass estimation. We find that this procedure is capable of recovering point estimates and confidence intervals of dynamical masses to a remarkably high degree of fidelity. The development of these uncertainty estimation techniques are a vital step towards constraining cosmology with deep learning cluster abundance measurements. Future work involving this method would investigate how more complex model inputs (e.g. 3D dynamical phase-space, multi-wavelength observations), finer tuning of hyperparameters (e.g. model architecture, KDE bandwidth), and alternative choices of variational weight distributions (e.g. Blundell et al. 2015) might improve recovery of mass posteriors. In addition, it will be important to study how mean-reversion biases for clusters on the low- and high-mass ends of our training catalog can be mediated in cluster abundance measurements.

We thank François Lanusse and Michelle Ntampaka for helpful conversations and feedback while developing this project. We thank Andrew Hearin and Peter Behroozi for preparing UniverseMachine catalogs of MDPL2 simulation data. This work is supported in part by DOE DE-SC0011114 and NSF 1563887. The computing resources necessary to complete this analysis were provided by the Pittsburgh Supercomputing Center. The CosmoSim database used in this paper is a service by the Leibniz-Institute for Astrophysics Potsdam (AIP). The MultiDark database was developed in cooperation with the Spanish MultiDark Consolider Project CSD2009-00064.

APPENDIX

Table 3. Explicit Loss Functions and Posteriors of Investigated Models

Model Name	$\mathcal{L}(\boldsymbol{\theta}, \mathcal{D})$	$p\left(m \mid \mathbf{x}, \boldsymbol{\eta}, \mathcal{D}\right)$
Point	$\frac{1}{2n}\sum_{i=1}^{n}\left(m_i - \mu\left(\mathbf{x}_i; \boldsymbol{\theta}\right)\right)^2 + \lambda\Vert\boldsymbol{\theta}\Vert_2^2$	$\mathcal{N}\left[m; \mu(\mathbf{x}; \hat{\boldsymbol{\theta}}), \sigma_{\mathcal{D}}^2\right]$
Point-d	$\frac{1}{2n}\sum_{i=1}^{n}\left(m_i - \mu\left(\mathbf{x}_i; \boldsymbol{\theta} \circ \mathbf{z}\right)\right)^2 + \lambda\Vert\boldsymbol{\theta}\Vert_2^2$	$\mathcal{N}\left[m; \hat{\mathbb{E}}_{\mathbf{z}}\left[\mu(\mathbf{x}; \hat{\boldsymbol{\theta}} \circ \mathbf{z})\right], \sigma_{\mathcal{D}}^2 + \hat{\mathrm{Var}}_{\mathbf{z}}\left[\mu(\mathbf{x}; \hat{\boldsymbol{\theta}} \circ \mathbf{z})\right]\right]$
Gauss	$\frac{1}{2n}\sum_{i=1}^{n}\left(\left[\left(m_i - \mu(\mathbf{x}_i; \boldsymbol{\theta})\right)/\sigma(\mathbf{x}_i; \boldsymbol{\theta})\right]^2 + \log\sigma(\mathbf{x}_i; \boldsymbol{\theta} \circ \mathbf{z})\right) + \lambda\Vert\boldsymbol{\theta}\Vert_2^2$	$\mathcal{N}\left(m; \mu(\mathbf{x}; \hat{\boldsymbol{\theta}}), \sigma(\mathbf{x}_i; \hat{\boldsymbol{\theta}})^2\right)$
Gauss-d	$\frac{1}{2n}\sum_{i=1}^{n}\left(\left[\left(m_i - \mu(\mathbf{x}_i; \boldsymbol{\theta} \circ \mathbf{z})\right)/\sigma(\mathbf{x}_i; \boldsymbol{\theta} \circ \mathbf{z})\right]^2 + \log\sigma(\mathbf{x}_i; \boldsymbol{\theta} \circ \mathbf{z})\right) + \lambda\Vert\boldsymbol{\theta}\Vert_2^2$	$\mathcal{N}\left(m; \hat{\mathbb{E}}_{\mathbf{z}}\left[\mu(\mathbf{x}; \hat{\boldsymbol{\theta}} \circ \mathbf{z})\right], \hat{\mathbb{E}}_{\mathbf{z}}\left[\sigma(\mathbf{x}; \hat{\boldsymbol{\theta}} \circ \mathbf{z})^2\right] + \hat{\mathrm{Var}}_{\mathbf{z}}\left[\mu(\mathbf{x}; \hat{\boldsymbol{\theta}} \circ \mathbf{z})\right]\right)$
Class	$-\sum_{i=1}^{n}\sum_{j=1}^{50}\left(\mathbb{I}[m_i \in \mathcal{C}_j]\ln g_j(\mathbf{x}_i; \boldsymbol{\theta}) + \mathbb{I}[m_i \notin \mathcal{C}_j]\ln\left[1 - g_j(\mathbf{x}_i; \boldsymbol{\theta})\right]\right) + \lambda\Vert\boldsymbol{\theta}\Vert_2^2$	$\text{Categorical}\left[m; S\left(\mathbf{g}(\mathbf{x}; \hat{\boldsymbol{\theta}})\right)\right]$
Class-d	$-\sum_{i=1}^{n}\sum_{j=1}^{50}\left(\mathbb{I}[m_i \in \mathcal{C}_j]\ln g_j(\mathbf{x}_i; \boldsymbol{\theta} \circ \mathbf{z}) + \mathbb{I}[m_i \notin \mathcal{C}_j]\ln\left[1 - g_j(\mathbf{x}_i; \boldsymbol{\theta} \circ \mathbf{z})\right]\right) + \lambda\Vert\boldsymbol{\theta}\Vert_2^2$	$\text{Categorical}\left[m; S\left(\hat{\mathbb{E}}_{\mathbf{z}}\left[\mathbf{g}(\mathbf{x}; \hat{\boldsymbol{\theta}} \circ \mathbf{z})\right]\right)\right]$

Note—Losses and posterior forms are identical for 1D and 2D variants of the above models. For clarity, the dependence of functional outputs μ, σ, and $\mathbf{g}$ on the fixed $\boldsymbol{\eta}$ have been suppressed. We train and evaluate our models using a labeled training dataset, $\mathcal{D} := \{(\mathbf{x}_i, m_i)\}_{i=1}^{n}$. For constant variance models, $\sigma_{\mathcal{D}}^2 = \frac{1}{n}\sum_{i=1}^{n}||m_i - \mu(\mathbf{x}_i; \hat{\boldsymbol{\theta}})||_2^2$. To express Dropout regularization, we define the stochastic variable $\mathbf{z} := \{z_i\}_{i=1}^{\dim(\boldsymbol{\theta})}$ where $z_i \sim \text{Bernoulli}(p_d)$. The operator $\circ$ represents element-wise multiplication. The operators $\hat{\mathbb{E}}_{\mathbf{z}}$ and $\hat{\mathrm{Var}}_{\mathbf{z}}$ represent empirical expectations and variances over $T = 100$ i.i.d. samples of $\mathbf{z}$, respectively. $S(\cdot)$ denotes the softmax function. C_j is the j-th bin of 50 regular classification bins spanning the mass range $13 \leq m \leq 16$.

REFERENCES

Abdullah, M. H., Wilson, G., & Klypin, A. 2018, ApJ, 861, 22, doi: 10.3847/1538-4357/aac5db

Allen, S. W., Evrard, A. E., & Mantz, A. B. 2011, ARA&A, 49, 409, doi: 10.1146/annurev-astro-081710-102514

Behroozi, P., Wechsler, R. H., Hearin, A. P., & Conroy, C. 2019, MNRAS, 488, 3143, doi: 10.1093/mnras/stz1182

Behroozi, P. S., Wechsler, R. H., & Wu, H.-Y. 2013, ApJ, 762, 109, doi: 10.1088/0004-637X/762/2/109

Bishop, M. A. 1994, Technical Report NCRG/94/004, Aston University. https://publications.aston.ac.uk/id/eprint/373/1/NCRG_94_004.pdf

Blundell, C., Cornebise, J., Kavukcuoglu, K., & Wierstra, D. 2015, arXiv e-prints, arXiv:1505.05424. https://arxiv.org/abs/1505.05424

Caldeira, J., & Nord, B. 2020, arXiv preprint arXiv:2004.10710

Calderon, V. F., & Berlind, A. A. 2019, MNRAS, 490, 2367, doi: 10.1093/mnras/stz2775

Carleo, G., Cirac, I., Cranmer, K., et al. 2019, Reviews of Modern Physics, 91, 045002, doi: 10.1103/RevModPhys.91.045002

Cook, S. R., Gelman, A., & Rubin, D. B. 2006, Journal of Computational and Graphical Statistics, 15, 675

Dodelson, S., Heitmann, K., Hirata, C., et al. 2016, arXiv e-prints, arXiv:1604.07626. https://arxiv.org/abs/1604.07626

Farahi, A., Evrard, A. E., Rozo, E., Rykoff, E. S., & Wechsler, R. H. 2016, MNRAS, 460, 3900, doi: 10.1093/mnras/stw1143

Farahi, A., Guglielmo, V., Evrard, A. E., et al. 2018, A&A, 620, A8, doi: 10.1051/0004-6361/201731321

Gal, Y. 2016, University of Cambridge, 1, 3

Gal, Y., & Ghahramani, Z. 2015, arXiv preprint arXiv:1506.02158

Gal, Y., & Ghahramani, Z. 2016, in international conference on machine learning, 1050–1059

Ho, M., Rau, M. M., Ntampaka, M., et al. 2019, ApJ, 887, 25, doi: 10.3847/1538-4357/ab4f82

Hoyle, B. 2016, Astronomy and Computing, 16, 34, doi: 10.1016/j.ascom.2016.03.006

Kendall, A., & Gal, Y. 2017, in Advances in neural information processing systems, 5574–5584

Kingma, D. P., & Ba, J. 2014, arXiv e-prints, arXiv:1412.6980. https://arxiv.org/abs/1412.6980

Klypin, A., Yepes, G., Gottlöber, S., Prada, F., & Heß, S. 2016, MNRAS, 457, 4340, doi: 10.1093/mnras/stw248

Kodi Ramanah, D., Wojtak, R., Ansari, Z., Gall, C., & Hjorth, J. 2020, arXiv e-prints, arXiv:2003.05951. https://arxiv.org/abs/2003.05951

Lanusse, F., Ma, Q., Li, N., et al. 2018, MNRAS, 473, 3895, doi: 10.1093/mnras/stx1665

LeCun, Y., Bengio, Y., & Hinton, G. 2015, nature, 521, 436

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

Mamon, G. A., Biviano, A., & Boué, G. 2013, MNRAS, 429, 3079, doi: 10.1093/mnras/sts565

Mantz, A. B., von der Linden, A., Allen, S. W., et al. 2015, MNRAS, 446, 2205, doi: 10.1093/mnras/stu2096

Möller, A., & de Boissière, T. 2020, MNRAS, 491, 4277, doi: 10.1093/mnras/stz3312

Neal, R. M. 2012, Bayesian learning for neural networks, Vol. 118 (Springer Science & Business Media)

Ntampaka, M., Trac, H., Sutherland, D. J., et al. 2015, ApJ, 803, 50, doi: 10.1088/0004-637X/803/2/50

—. 2016, ApJ, 831, 135, doi: 10.3847/0004-637X/831/2/135

Ntampaka, M., ZuHone, J., Eisenstein, D., et al. 2019, ApJ, 876, 82, doi: 10.3847/1538-4357/ab14eb

Old, L., Wojtak, R., Pearce, F. R., et al. 2018, MNRAS, 475, 853, doi: 10.1093/mnras/stx3241

Planck Collaboration, Ade, P. A. R., Aghanim, N., et al. 2014, A&A, 571, A16, doi: 10.1051/0004-6361/201321591

—. 2016, A&A, 594, A24, doi: 10.1051/0004-6361/201525833

Scott, D. W. 2015, Multivariate density estimation: theory, practice, and visualization (John Wiley & Sons)

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. 2014, Journal of Machine Learning Research, 15, 1929. http://jmlr.org/papers/v15/srivastava14a.html

Voit, G. M. 2005, Reviews of Modern Physics, 77, 207, doi: 10.1103/RevModPhys.77.207

Wojtak, R., Łokas, E. L., Mamon, G. A., et al. 2007, A&A, 466, 437, doi: 10.1051/0004-6361:20066813

Wojtak, R., Old, L., Mamon, G. A., et al. 2018, MNRAS, 481, 324, doi: 10.1093/mnras/sty2257

Zoph, B., & Le, Q. V. 2016, arXiv preprint arXiv:1611.01578

Zwicky, F. 1933, Helvetica Physica Acta, 6, 110