Parametric Density Estimation with Uncertainty using Deep Ensembles

A.L. Peirson [*]
Kavli Institute for Particle Astrophysics and Cosmology
Stanford University
Stanford, CA 94305
alpv95@stanford.edu

Taylor Howell
Department of Mechanical Engineering
Stanford, CA 94305
thowell@stanford.edu

Marius Tirlea
Department of Statistics
Stanford, CA 94305
mtirlea@stanford.edu

Abstract

In parametric density estimation, the parameters of a known probability density are typically recovered from measurements by maximizing the log-likelihood. Prior knowledge of measurement uncertainties is not included in this method – potentially producing degraded or even biased parameter estimates. We propose an efficient two-step, general-purpose approach for parametric density estimation using deep ensembles. Feature predictions and their uncertainties are returned by a deep ensemble and then combined in an importance weighted maximum likelihood estimation to recover parameters representing a known density along with their respective errors. To compare the bias-variance tradeoff of different approaches, we define an appropriate figure of merit. We illustrate a number of use cases for our method in the physical sciences and demonstrate state-of-the-art results for X-ray polarimetry that outperforms current classical and deep learning methods.

1 Introduction

The application of deep neural networks (NNs) is ubiquitous across a wide variety of domains such as computer vision, NLP, audio, and the physical sciences. The majority of state-of-the-art NN performances are single (high-dimensional) input, multiple-output tasks, for instance classifying images [17], scene understanding [26] and voice recognition [9]. These tasks typically involve one input vector or image and a single output vector of predictions.

In parametric density estimation, there is a known probability density (distribution) that the data (or latent features of the data) follow. The goal is to find representative distribution parameters for a given dataset. In simple cases where the likelihood is calculable, maximum likelihood estimation can be used effectively. In cases where latent features of the data follow a known distribution (e.g., heights of people in a dataset of photographs), NNs can potentially be used to directly estimate the distribution parameters. For clarity, we define this direct/end-to-end approach as parametric feature density estimation (PFDE). Such an approach, requires employing entire datasets (with potentially thousands to millions of high-dimensional examples) as inputs in order to output a vector of density parameters. Furthermore, to be useful these NNs would need to generalize to arbitrarily sized dataset-inputs.

[*]https://www.alpeirson.com

One example of NNs making sense of large dataset-inputs is found in natural language processing. Here large text corpora, converted to word vectors [25, 5], can be input and summarized by single output vectors using recurrent neural networks (RNNs), for instance in sentiment analysis [3]. However, these problems and RNNs themselves contain inductive bias – there is inherent structure in text. Not all information needs be given at once and a concept of memory or attention is sufficient [31]. The same can be said about time domain problems, such as audio processing / voice recognition. Memory is inherently imperfect – for PFDE, one ideally wants to know all elements of the ensemble at once to make the best prediction; sequential inductive bias is undesirable. Ultimately, memory and architectural constraints make training NNs for general direct PFDE computationally intractable.

On the other hand, density estimation on data directly (not on its latent features), is computationally tractable. Density estimation can let us find a complete statistical model of the data generating process. This is a powerful tool - applying deep learning to density estimation has advanced the field significantly [22]. Most of the work so far focuses on density estimation where the density is unknown *a priori*. This can be achieved with non-parametric methods such as neural density estimation [23], or with parametric methods such as mixture density networks [2]. In PFDE, however, we have a known probability density over some features of the whole dataset. The features may be more difficult to predict accurately in some datapoints than others.

Typical parametric density estimation does not make use of data uncertainties where some elements in the dataset may be more noisy than others. Not including uncertainty information can lead to biased or even degraded parameter estimates. The simplest example of parametric density estimation using uncertainties is a weighted mean. This is the result of a maximum likelihood estimate for a multi-dimensional Gaussian. For density estimation on predicted data features, PFDE, we would like a way to quantify the predictive uncertainty. This problem has received a lot of recent attention in deep learning, with many NN frameworks providing predictive uncertainties. Bayesian NNs model their weights as a distribution with a prior so that given training data, the posterior over the weights (and thus the predictions) can be computed [20]. In practice, this means using regularization as a prior and dropout to sample the posterior [14]. In general these are hard to implement, slow to train, and produce predictive uncertainties that are highly dependent on the chosen prior. A more general solution is offered by deep ensembles [18]. While these are not strictly equivalent to a Bayesian approach, although they can be made such using appropriate regularization [24], they offer practical predictive uncertainties and have been shown to generalize more readily [8].

In this work, we propose a NN approach that circumvents large dataset-input training or recurrent architectures to predict known feature density parameters over large input datasets. We use predictive uncertainties on features of individual dataset elements as importance weights in a maximum likelihood estimation. We will show that estimating known density parameters in a 2-step approach provides greater interpretability and flexibility. We are able to predict uncertainties on our density parameter estimates using parametric bootstrap methods [6]. Our method is widely applicable to a number of applied machine learning fields; §3 showcases a few important examples.

Contributions: Our contributions in this paper are as follows: (1) We introduce a general, flexible method for PFDE using NNs. The method can be applied to any domain requiring PFDE. We illustrate a number of varied domain examples in the physical sciences in §3. (2) In an in-depth evaluation we show that our method outperforms not only classical methods for density estimation, but also standard NN implementations in an application to X-ray polarimetry. (3) We investigate the bias-variance tradeoff associated with our method and introduce a tuneable hyperparameter to control it. *Note:* In the following we focus on regression examples, (since unbinned density estimation is preferable to binned). However, a similar method can be applied to prediction examples where softmax class probabilities are used as heteroscedastic aleatoric uncertainty.

2 Importance weighted estimation with deep ensembles

In this section we propose predicting known feature density parameters over input datasets using an importance weighted maximum likelihood estimate informed by deep ensemble outputs. We outline the basic problem and high level algorithm in §2.1. We describe the practical training of deep ensembles with modified ensemble objectives, including the advantages for this approach in §2.2. In §2.3 and §2.4 we describe the modified log-likelihood used to solve for the final density parameters, and how to solve it in both convex and non-convex cases. Algorithm 1 outlines the full method.

2.1 Problem setup and high-level summary

We wish to estimate the feature density parameters of K high dimensional data points $\{\mathbf{x}\}$: $f(\{\mathbf{x}_n\}_{n=1}^{K})$. Here $\mathbf{x} \in \mathbb{R}^D$ can be any high dimensional data (e.g. images, time series) and K is arbitrary. For example, consider estimating the mean and standard deviation of human heights from a dataset consisting of photographs of people. A person's height in each photograph is the image feature and we know this feature approximately follows a Gaussian distribution.

In general, the function f mapping the high dimensional data points to the desired density parameters is unknown, since the high dimensional data is abstracted from its features. While learning f directly is typically infeasible, as mentioned in §1, the function g mapping data features y_n to the density parameters is known.

We cast this as a supervised learning problem where we have a dataset D consisting of N data points $D = \{\mathbf{x}_n, y_n\}_{n=1}^{N}$ with labels $y \in \mathbb{R}^M$ where $\mathbf{x} \in \mathbb{R}^D$. We want to estimate the density parameters for an unseen test set $g(\{y_n\}_{n=1}^{K})$ for arbitrary K.

The basic recipe that comes to mind is training a single NN to predict output labels $\{y_n\}_{n=1}^{K}$ then evaluate g directly. This ignores the high variance in single NN predictions (dependent on training/random initialization), that some individual examples may be more informative than others, and that an objective to predict the most accurate output labels may not be the best for predicting good density parameters (high bias may be introduced, for instance).

Our hybrid approach is as follows. (i) Train a deep ensemble of NNs[2] to predict $\{y_n \pm \sigma_n\}_{n=1}^{K*N}$ where N is the number of NNs in the ensemble and σ_n is the aleatoric uncertainty of each prediction, (ii) select the best M networks from the ensemble using a modified objective specific to the density parameters to be estimated, (iii) use the $\{\sigma_n\}_{n=1}^{K*M}$ as weights in an importance weighted maximum likelihood estimate. The next section, §2.2, describes procedures (i) and (ii).

2.2 Deep ensembles

Deep ensembles [18] return robust and accurate supervised learning predictions and predictive uncertainties, which enable the best density parameter predictions. These use an ensemble of individual NNs (with different random initializations) trained to predict features and their statistical uncertainties. Final predictions and their systematic uncertainties are then recovered by combining the estimates from each of the NNs in the ensemble.

There are two germane types of uncertainty one can model [14]. Aleatoric uncertainty captures noise inherent in the observations. This is irreducible given the data, for example, like sensor noise. This is equivalent to statistical uncertainty in the physical sciences. On the other hand, epistemic uncertainty accounts for uncertainty in the model parameters – uncertainty which captures our ignorance about which model generated our collected data. This uncertainty can be explained away given enough data, and is often referred to as model uncertainty or systematic uncertainty in the physical sciences. If aleatoric uncertainty is variable for different datapoints, it is known as heteroscedastic uncertainty, i.e., some inputs to the model potentially have more noisy outputs than others. This is common in computer vision and very important in PFDE, since some datapoints will be much more informative than others in estimating density parameters.

In regression, deep ensembles model heteroscedastic aleatoric uncertainty by modifying the typical mean-squared errors (MSE) objective to a negative log-likelihood (NLL) [18],

$$\mathrm{Loss}(\mathrm{y}|\mathbf{x}) = \frac{1}{2}\log\sigma^2(\mathbf{x}) + \frac{1}{2\sigma^2(\mathbf{x})}\|\mathrm{y} - \hat{\mathrm{y}}(\mathbf{x}))\|_2^2. \tag{1}$$

Extensions using more complex distributions like mixture density networks or heavy tailed distributions may be more applicable to certain problems with prior knowledge about the error distribution.

Epistemic uncertainty is modelled using a uniformly weighted ensemble of M NNs each trained starting from a different random initialization. The prediction from each of these ensembles is approximated as a Gaussian distribution - the regression prediction and uncertainty are given by the mean and standard deviation respectively (each NN in the ensemble contributes equally). The epistemic uncertainty is then typically combined with the aleatoric in quadrature. Typically $M \sim$

[2]We note that the NN architecture used will of course depend on the dataset domain.

$5 - 15$. Epistemic uncertainty is handled in part (iii) of our approach, where we avoid Gaussian assumptions about the epistemic uncertainty and allow for non-uniform NN contribution in the ensemble.

In part (i) of our hybrid approach for PFDE, we train a deep ensemble to minimize the NLL (1) on desired features. We follow the deep ensemble training procedure outlined in [18] (with recast loss function from [14]) without using adversial examples, using the full dataset for each NN, etc. Since the individual density parameters over predicted features are the final desired values in PFDE, it is unlikely that an objective maximizing feature accuracy on the validation set is the true objective. Even if feature prediction accuracy on individual datapoints $\{\mathbf{x}_n, y_n\}$ were maximized, prediction biases may be introduced that make for skewed density parameters. The high parameterization, finite data-dependence and non-linear nature of NNs make this very likely. The single CNN method in fig. 2, §3.4, shows a very clear case. We have identified two ways of resolving this issue:

1. Include terms in the individual NN objectives to penalize known sources of bias.
2. Select the top M performing NNs, as measured by a criterion that includes density parameter prediction bias on a held out test set.

In practice both can be used simultaneously. However, the former runs into batch size problems (since one needs a large sample size to accurately estimate bias), and the source of bias is not always well understood. We have found the latter to be sufficient and a simpler solution, naturally arising from the use of deep ensembles. A good NN selection criterion balances MSE on predicted features with bias on the predicted density parameters. The exact criterion is usually domain specific, depending on the density to be estimated. We give a concrete example of a selection criterion in our applications §3. NN selection for the final ensemble to minimize the bias in density parameter predictions is part (ii) of our PFDE method.

2.3 Importance weighted log-likelihood

Provided a mapping between high dimensional inputs and interpretable features $\mathbf{x_n} \mapsto y_n$, we can calculate the density parameters $\psi_1, \psi_2, ...\psi_k$ by minimizing the appropriate negative log-likelihood function $p(\{y_n\}|\psi_1, \psi_2, ...\psi_k)$. Some feature predictions y_n will have greater aleatoric, σ_n, and epistemic predictive uncertainties. We estimate feature density parameters by incorporating both of these uncertainties into an importance weighted maximum likelihood estimate. This makes up part (iii) of our hybrid method.

An importance weight quantifies the relative importance of one example over another. Importance weighting an element should be the same as if that element were included multiple times in the dataset, proportional to its importance weight [13]. The deep ensemble, once trained and the top M NNs selected, will act as mapping between high dimensional inputs $\mathbf{x}_n$ and feature-uncertainty output pairs y_n, σ_n. For each input $\mathbf{x}_n$ there will be M output pairs $\{y_{nm}, \sigma_{nm}\}_{m=1}^{M}$, one for each NN in the deep ensemble. In order to use all possible information when estimating the desired density parameters $\psi_1, \psi_2, ...\psi_k$, we define an importance weighted negative log-likelihood function

$$L_{\mathrm{w}}(\{y_n\}, \psi_1, \psi_2, ...\psi_k) = -\sum_{m=1}^{M}\sum_{n=1}^{N} w_{nm} \mathrm{log}\mathcal{L}(\mathrm{y_{nm}}|\psi_1, \psi_2, ...\psi_\mathrm{k}) \tag{2}$$

$$w_{nm} = b_m . \sigma_{nm}^{-\lambda} \tag{3}$$

For a data sample of size N, each of the M NNs produces N predictions. These are labelled y_{nm}. Rather than averaging the output features over the NNs in the ensemble as in [18], we can simply include all of them in L_w, so there are a total of $M * N$ feature inputs. This will properly account for the epistemic uncertainty in each y_n by including its full distribution – it does not make any assumptions about the shape (being Gaussian).

Each individual prediction y_{nm} has an associated importance weight w_{mn}. The weights $\{b_m\}_{m=1}^{M}$ correspond to the NNs. These can be uniform, $b_m = 1 \forall m$, for predictions from each NN in the ensemble to be weighted equally, or they can be a function of the NN objective. The $\sigma_{nm}^{-\lambda}$ term weights each y_{nm} by its predictive uncertainty. The hyperparamter $\lambda \geq 0$ controls the importance weighting distribution. A high λ means the most "certain" y_{nm} will dominate the final ensemble statistic. As always in estimation problems, there is a trade-off between lower variance predictions and more bias. This can be tuned for a specific application using λ; we discuss the procedure in detail in our example application, §3. Final density parameters are found by minimizing (2) over the domain of the density parameters ψ.

Typically, the weights in weighted likelihood estimation are determined heuristically [12]. In this example, we choose $w = \sigma^{-\lambda}$ since it approximates the simple functional form of the likelihood used in a weighted mean estimate ($\lambda = 2$). This weighting choice is also inspired by the dispersion parameter used in generalized linear models (GLMs) [21]. We expect that this weighting will retain similar robustness properties in terms of model fitting, and will generalize well to many domains. However, of course, any decreasing function $f : \mathbb{R}^+ \to \mathbb{R}^+$ may be used to determine weights, with the most suitable choice of function f within a given class of functions (in our case, parametrized by λ) to be determined by either cross-validation or performance on a holdout set. Further discussion of weight choice in our application is given in section §3.4.

Confidence intervals on the density parameters can be calculated using the parametric bootstrap [6]: select N y_{nm}, σ_{nm} pairs with replacement and minimize eq. 2. In the limit of many trials with different random subsamples, this will give the output distribution on the density parameters.

2.4 Density parameter regression

For a special class of parameterized densities it is possible to find the global minimizer or minimize (2) analytically (e.g. for a multivariate Gaussian). In practice, the majority of parametric densities of interest for PFDE are likely to be convex (exponential families, our application example §3, etc.), so will fall into this special class. In the general case, minimization is performed numerically to find locally optimal solutions.

In this work, we employ Ipopt [34], an open-source interior-point solver for large-scale non-convex optimization problems, to minimize (2). This method can be used for convex or non-convex parametric density estimates, but only convex ones are guaranteed to be global optimal. Because Ipopt finds locally optimal solutions, which are highly dependent upon an initial guess of the parameters provided to the solver, in the non-convex case, we recommend nested sampling [7] to test many initial guesses and then select the best local solution. Constraints on the density parameters, for instance if they have a finite domain, can be incorporated for both the convex and non-convex case.

The overall training and evaluation procedure is summarized in Algorithm 1.

Algorithm 1: Pseudocode for our PFDE method.

```
1: Identify output features $y_n$ relevant to the desired density parameter(s) (e.g., subject height in
   photographs).
2: Train a deep ensemble of NNs using loss function (1) to maximise accuracy on the desired
   output features
for nn in NNs do
    3: Evaluate the density parameter(s) on a test dataset by minimizing
       $\sum_{n=1}^{N} \log\mathcal{L}(\mathrm{y_n}|\psi_1, \psi_2, ...\psi_\mathrm{k})$ w.r.t. $\psi_1, \psi_2, ...\psi_k$
end
4: Select the $M$ top performing NNs.
5: Evaluate the density parameter(s) using importance weights by minimizing (2). (optional:
   weight NNs by their performances in step 3).
6: Tune $\lambda$ hyperparameter for the specific application.
```

3 Experiments

In this section we illustrate the described method with some specific examples related to the physical sciences and engineering.

3.1 X-ray polarimetry

Measuring X-ray polarization has been a major goal in astrophysics for the last 40 years. X-ray polarization can provide essential measurements of magnetic fields very close to high energy sources, such as accreting black holes and astrophysical jets [32]. The recent development of photoelectron

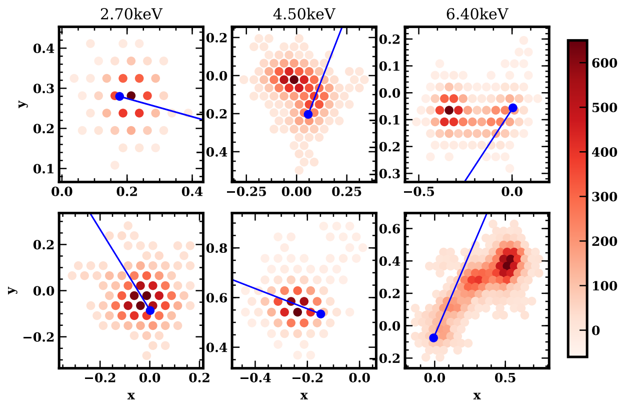


Figure 1: Example IXPE track images at 2.7, 4.5 and 6.4 keV energies (columns). The blue lines show the initial photoelectron direction. Color represents the amount of charge deposited in a hexagonal pixel. Track morphology (and thus angle reconstruction) depends strongly on energy.

tracking detectors [1] has greatly improved the prospects of doing so. X-ray polarization telescopes with photoelectron tracking detectors directly image electron tracks formed from photoelectrons scattered by the incoming X-ray photons. This technique has the capability to build up an of image extended sources. We describe an application of our hybrid PFDE method to X-ray polarimetry using photoelectron tracking detectors. We use data from the upcoming NASA Imaging X-ray Polarization explorer (IXPE) [29] as a working example. The problem of recovering polarization parameters from a dataset of (IXPE) electron track images has recently been announced as an open problem in the machine learning community [19].

3.1.1 Problem setup

The linear polarization of light can be fully described by two degrees of freedom: the polarization fraction $0 \le \Pi \le 1$, (0% – 100%), and the electric vector position angle $-\pi/2 \le \phi \le \pi/2$. These can be thought of as the magnitude and direction of a vector perpendicular to the direction of propagation of the light. We restrict $-\pi/2 \le \phi \le \pi/2$, due to an ambiguity in orientation (the polarization vector is two-headed). $\Pi = 1$ means the light is 100% polarized. In imaging X-ray polarimetry, when the detector is pointed at an X-ray source, it measures individual 2D images of electron tracks. The initial directions the electrons travel follow a known probability density that depend on the source polarization, and the problem is to recover the polarization parameters Π and ϕ from this dataset of 2D track images.

In the case of IXPE, X-ray photons are directed into a detector filled with gas where they knock electrons off the gas molecules. These freed electrons (photoelectrons) leave charge tracks that are imaged by the hexagonal pixels of the detector. Fig. 1 shows some example photoelectron tracks at different X-ray energies. Each track represents the interaction of a single photon with a single gas molecule. The ejection direction θ of photoelectrons follows the probability density

$$p(\theta \mid \Pi, \phi) = \frac{1}{2\pi}\big(1 + \Pi\cos\big(2(\theta + \phi)\big)\big) \ , \tag{4}$$

where Π and ϕ are fixed polarization parameters that depend on the source. The density depends only on the polarization properties of the incoming X-ray photons. Hence, by estimating θ for a large number of tracks, we may recover the original polarization parameters Π and ϕ, using parametric density estimation.

Track morphologies vary greatly with energy (and even for the same energy); this affects how difficult it is to recover an accurate intial photoelectron angle θ. Low energy tracks are typically less elliptical and so more difficult to estimate. IXPE's effective energy range is $\sim 2 - 8$ keV. However, since most astrophysical sources display negative power law distributions in X-ray flux, and the effective area of the telescope is much higher at low energy, the vast majority of measured tracks will be in the $\sim 2 - 4$ keV range. For this reason it is essential to incorporate some form of quality control in the tracks used for polarization estimates.

Current IXPE methods estimate individual track θ using a moment analysis [28]. This calculates the first, second and third charge moments using the (x, y) coordinates of the hexagonal detector pixels, combining them to extract θ. For each track, a single $-\pi \leq \theta \leq \pi$ is output. The polarization parameters are then estimated from the $\{\theta\}$ ensemble using an unbinned sufficient statistic [15], equivalent to minimizing the unweighted log-likelihood. The moment analysis additionally outputs an estimate of the track ellipticity, which can be used as a proxy for θ estimation accuracy. The standard moment analysis uses a track cut to improve polarization recovery – 20% of the tracks are cut based on ellipticity, a suboptimal solution since a large fraction of the data is thrown away. NNs have also recently been applied to this problem [16]. This approach uses single CNNs for classification on θ, with binned fits to θ histograms to extract polarization parameters and track quality cuts. Our hybrid method exhibits significantly improved performance over this basic NN approach.

3.2 Parametric feature density estimation

Following §2, we define CNNs that take single track images as input and (θ, σ_θ) as output. In this case the track angles θ are the data features that follow the known distribution (4), and the CNNs will make up the deep ensemble.

To make the hexagonal track images useable as inputs to typical CNN architectures, we first convert the hexagonal images to square image arrays by shifting every other column and rescaling the distance between points, as described in [30]. Since there are two possible shifts (odd and even rows), we apply both and stack the two shifted images, like color channels in rgb images, partly because the CNN convolutional kernels are not spatially equivariant in hexagonal space. We find this reduces NN density parameter prediction bias.

To recover Π, ϕ we need to predict 2θ, so we use the loss function (1) but parameterize the true angle θ as a 2D vector $\mathbf{v} = (\cos 2\theta, \sin 2\theta)$ to capture the periodicity. The loss function is as follows:

$$\text{Loss}(\mathbf{v}, \hat{\mathbf{v}}) = \frac{1}{2}\log\sigma^2 + \frac{1}{2\sigma^2}\|\mathbf{v} - \hat{\mathbf{v}}\|_2^2 \tag{5}$$

Each of the NNs in the ensemble output the 3-vector $(\hat{\mathbf{v}}, \sigma)$. Then $\hat{\theta} = \arctan\frac{\hat{v}_2}{\hat{v}_1}/2$. To calculate the final Π, ϕ with an ensemble of M NNs for a given test dataset with N tracks we minimize the importance weighted NLL (2) with likelihood

$$L(\{\hat{\theta}\}|\Pi, \phi) = \frac{1}{2\pi}(1 + \Pi\cos(2(\hat{\theta}_{\text{nm}} + \phi))). \tag{6}$$

We can recast this as the convex optimization problem

$$\begin{array}{ll} \underset{\mathbf{x}}{\text{minimize}} & -\sum\limits_{m=1}^{M}\sum\limits_{n=1}^{N} b_m.\sigma_{nm}^{-\lambda}\log\left(1 + \mathbf{v}_{nm}^T\mathbf{x}\right) \\ \text{subject to} & \|\mathbf{x}\|_2 \leq 1 \end{array} \tag{7}$$

where $\mathbf{v}_{nm} = (\cos\theta_{nm}, \sin\theta_{nm})$ and $\mathbf{x} = (\Pi\cos\phi, \Pi\sin\phi)$. By recasting (2) as a convex optimization problem, we have a guaranteed globally optimal solution for (Π, ϕ). We can solve (7) quickly and efficiently using second order Newton methods. In practice we use the robust open source software IpOpt, §2.4.

3.2.1 Figure of merit: minimum detectable polarization

In polarization estimation, we want high recovered $\hat{\Pi}_{100\%}$ (and accurate ϕ) for a known 100% polarized source ($\Pi_{\text{true}} = 1$), and low recovered $\Pi_{0\%}$ for an unpolarized source ($\Pi_{\text{true}} = 0$). Since there is irreducible noise in the tracks, it is impossible for any method to achieve $\Pi_{100\%} \sim 1$, so $\hat{\Pi}_{\text{meas}}$ estimates are calibrated to get the final $\hat{\Pi}$ for an unknown source[3]: $\hat{\Pi} = \hat{\Pi}_{\text{meas}}/\hat{\Pi}_{100\%}$. The minimum detectable polarization (MDP) may then be defined as

$$\text{MDP}(N) = \hat{\Pi}_{0\%}/\hat{\Pi}_{100\%}. \tag{8}$$

The MDP is a function of N, the number of track images. We use the MDP as a figure of merit to evaluate model performance: a lower MDP (for a given N) means better polarization estimation. This

[3] $\Pi_{100\%}$ is measured before on a source with the same track energy distribution.

is effectively a measure of the signal to noise ratio and is a more robust metric than the MDP typically defined for X-ray polarization [33], since it does not preclude biased estimators. It is evaluated on unseen polarized and unpolarized datasets. In estimating the MDP, we take the largest possible N given our limited heldout test data ($N \sim 300,000$). We use the MDP as the criterion to select the hyperparameter λ in (2). In this way we can tradeoff accuracy and bias in our Π, ϕ estimates.

3.3 NN Training and selection

Our intial dataset consists of 3 million simulated tracks, examples of which are shown in fig. 1. The track energies uniformly span $1.8 - 8.2$keV, IXPE's most sensitive range and are unpolarized (uniform track angle distribution). Since we don't know a priori what energy each track is, we want NNs that can make predictions for tracks of all energies. This also makes for a more generalizable system, since some high energy tracks have similar characteristics to lower energy ones.

Each track is labelled with its 2D angle vector. We split these 3 million tracks into a training set and a validation set, where the validation set makes up $\sim 5\%$ of the total. Additionally, we have simulated track test sets (polarized and unpolarized). These are used in the net selection process (~ 300 thousand tracks, at a range of energies).

We use a ResNet-19 [11] convolutional NN architecture as our base NN. This particular architecture is large enough to overfit the training set, and trains in a reasonable amount of time ($\sim 12 - 20$ GPU hours). Before training we preprocess the training data (square track images). We apply pixelwise centering and rescaling. This subtracts the pixelwise mean from each track image and divides by the pixelwise standard deviation (where the mean and standard deviation are calculated over the full training set). We use stochastic gradient descent with momentum and a decaying learning rate starting at $1e-2$. We choose batch sizes between $512 - 4096$ (tracks per batch). We trained for 70 epochs, saving NN checkpoints every 10 epochs for use in the NN selection, §3.3.1. We use L_2-norm regularization 5×10^{-5}.

It is important to recall that the training procedure minimizes the NLL (5), but does not minimize the error on polarization, Π, ϕ, estimation. The next section, §3.3.1, covers network selection for the ensemble to get the best polarization estimates.

3.3.1 Selection

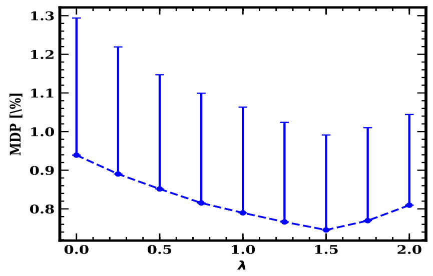
To select NNs for the deep ensemble, we train a number ($\sim 25 - 40$) of individual NNs in the manner described above. We use the entire training dataset to train each NN. After training these individual NNs, we downselect to the best performing $\sim 10 - 15$, as measured by a quality metric. Since we save NN checkpoints throughout the training, we consider NNs at all middle-late stages of their training in our selection. This is a form of early-stopping to prevent over-fitting on the training set. We define our quality metric as

$$Q = \prod_{n=1}^{N} \mathrm{MSE}_n * \Pi_{0\%_n}. \tag{9}$$

This balances the MSE on individual track angle prediction with the recovered polarization fraction Π for unpolarized test datasets over a range of energies n: 2.7, 3.7, 4.5, 5.9, 6.4, 7.5keV. MSE_n corresponds to recovered signal at energy n and $\Pi_{0\%_n}$ corresponds to recovered noise/bias. Lower values for Q are better. By using an ensemble of NNs selected by the quality factor, (9), we maximize the signal while averaging out the noise. The results in the next section use an ensemble of $M = 14$ networks; each network is weighted uniformly, $b_m = 1 \forall m$ in (3,7).

3.4 Results

The right panel of fig.2 shows the results of our deep ensemble PFDE method alongside the current state of the art methods. The single CNN method with optimized cuts, developed in [16], provides significant improvements in $\Pi_{100\%}$ over the moment analysis, but adds bias to the unpolarized measurement, increasing its MDP and making it a worse method for the majority of energies. Our method, with λ tuned accordingly for each energy, outperforms the rest especially in the power law datasets, where there is an almost $\sim$ 2x reduction in MDP. This shows the power of a simple weighted scheme over quality cuts in PFDE, it allows our method to take advantage of higher signal ($\Pi_{100\%}$) at higher energies in the power law datasets. The λ tuning procedure is shown in the left panel of fig.2.



Energy	Moments	Mom. w/ cuts	Single CNN w/ cuts [16]	Our Method	
	MDP [%] (68% CI)	MDP [%]	MDP [%]	MDP [%]	λ
2.7 keV	2.3 (2.7)	2.0 (2.3)	2.6 (3.3)	**1.9 (2.2)**	2.75
4.5 keV	1.0 (1.2)	0.9 (1.2)	1.5 (1.9)	**0.9 (1.2)**	2.5
6.4 keV	0.8 (0.9)	0.8 (1.0)	1.6 (1.9)	**0.6 (0.8)**	1.25
8.0 keV	0.8 (1.0)	0.7 (0.9)	0.8 (1.1)	**0.7 (0.8)**	0.5
PL 1	1.0 (1.3)	0.8 (1.2)	n/a	**0.5 (0.7)**	1
PL 2	1.1 (1.5)	1.1 (1.4)	n/a	**0.7 (1.0)**	1.5

Figure 2: *Right:* Results on energy selected track image datasets, comparing our method with the current state of the art. PL1 and PL2 are power law datasets with range spanning $1.8 - 8.2$keV (PL1 $dN/dE \propto E^{-1}$, and PL2 $dN/dE \propto E^{-2}$). Individual energy datasets have 300 thousand tracks each, PL datasets have 1 million. All methods have RMSE$_\phi \leq 0.5°$. The weighting hyperparameter λ decreases with energy since the statistical error σ distribution is more closely clustered for lower energies. *Left:* MDP as a function of hyperparameter λ for the PL2 dataset. This method is used to select all of the λ. Confidence intervals (68%) are calculated using the parametric bootstrap in both panels; for MDP only the upper bound is necessary.

We plan to release further results and more domain specific information for this particular application in a subsequent publication.

3.5 Other applications

There are numerous application of PFDE with uncertainty in the physical sciences and engineering. In high energy particle physics massive, short-lived particles can be detected by fitting a Cauchy distribution to the frequencies of measured decay states. Raw sensor data from hadronic particle colliders like the LHC are very noisy with variable uncertainty, meaning our PFDE approach to estimate the Cauchy distribution parameters could be very fruitful. This especially true with the widespread current use of deep learning in particle physics [10]. Our approach is heuristically justified due to the asymptotic efficiency of the maximum likelihood estimator in a Cauchy location model [4]. In manufacturing, GLMs fit to binomial distributions are commonly used to assess product quality, or the probability of a product being defunct. Today, computer vision is used for much of the inspection [27], making our hybrid PFDE method a potential step forward. These are just a few application examples – our method may be useful for any GLM based method with high dimensional data.

4 Discussion

We have proposed a supervised learning framework for parametric feature density estimation. Our method uses deep ensembles to predict high dimensional data features, their aleatoric and epistemic uncertainties. We estimate feature density parameters by incorporating both of these uncertainties into an importance weighted maximum likelihood estimate. To minimize the bias in density parameter predictions, we select NNs for the ensemble using a quality metric that balances the MSE of feature estimates with bias in feature parameter predictions. We also include a tuneable weighting hyperparameter λ, allowing one to control the bias-variance tradeoff for density estimation. Intuitively, in many real feature density estimation problems, some high dimensional data points may be much more informative than others due to complex noise or differing generative distributions. Our method models this explicitly, weighting datapoint features by their predictive uncertainty when estimating density parameters. This avoids throwing away valuable data with quality cuts, yielding improved density estimates. Our method is scaleable to any feature dataset size and is completely flexible for specific domain applications; most NN architectures can be used. We achieve state-of-the-art results over standard deep learning methods and classical algorithms in X-ray polarimetry - a recent open problem in ML. We expect our method would provide similar improvements to a number of PFDE application fields, including high energy particle physics and manufacturing. While our method works well for densities with convex log-likelihoods, non-convex ones will not necessarily yield globally optimal solutions and may be very time consuming to evaluate. *Future Work*: Future additions to the method include more complex aleatoric uncertainty modelling. We assume a Gaussian distribution for our feature prediction (1), but for domain applications where there is an expected feature uncertainty, one could use an alternative distribution, or even a mixture density network [2] for more flexibility. In that case the functional form of weighting would have to be reconsidered. Non-uniform weightings for different NNs in the deep ensemble and their effect on prediction bias could also be explored. A better understanding of prediction bias in single NNs would be very useful.

4.1 Broader Impact Statement

Use of AI techniques in both scientific research and applied engineering is becoming the status quo. There is an increasing demand for simple, scaleable approaches to solve a variety of data-centered problems. In this paper, we describe a method combining parametric and non-parametric methods for density estimation with uncertainty. We have shown that our simple approach can improve density estimation in applications with variable noisy data. We have already shown state-of-the-art results on an open problem in the physical sciences: X-ray polarimetry. Our method can improve the sensitivity of the upcoming NASA telescope IXPE by almost 2 times, reducing the required observing time for individual sources by the same factor. This will vastly expand the amount of science that can be done in the telescope's limited lifetime. Since our method uses deep ensembles to predict features and their predictive uncertainties, by choosing the appropriate NN architecture the method becomes generalizable to many deep learning domains. We have suggested a few, including manufacturing and experimental physics, where our method is likely to provide improvements similar to IXPE. We expect that our method will have a beneficial impact in many different subject areas.

References

[1] Ronaldo Bellazzini, F. Angelini, Luca Baldini, Alessandro Brez, Enrico Costa, Giuseppe Di Persio, Luca Latronico, M. M. Massai, Nicola Omodei, Luigi Pacciani, Paolo Soffitta, and Gloria Spandre. Novel gaseous x-ray polarimeter: data analysis and simulation. In *Polarimetry in Astronomy*, volume 4843, pages 383–393. International Society for Optics and Photonics, February 2003.

[2] Christopher Bishop. Mixture Density Networks. January 1994.

[3] Ethem F. Can, Aysu Ezen-Can, and Fazli Can. Multilingual Sentiment Analysis: An RNN-Based Framework for Limited Data. *arXiv:1806.04511 [cs]*, June 2018. arXiv: 1806.04511.

[4] Gabriela V. Cohen Freue. The Pitman estimator of the Cauchy location parameter. *Journal of Statistical Planning and Inference*, 137(6):1900–1913, June 2007.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805 [cs]*, May 2019. arXiv: 1810.04805.

[6] B. Efron. Bootstrap Methods: Another Look at the Jackknife. *Annals of Statistics*, 7(1):1–26, January 1979. Publisher: Institute of Mathematical Statistics.

[7] F. Feroz, M. P. Hobson, and M. Bridges. MultiNest: an efficient and robust Bayesian inference tool for cosmology and particle physics. *Monthly Notices of the Royal Astronomical Society*, 398(4):1601–1614, October 2009. arXiv: 0809.3437.

[8] Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep Ensembles: A Loss Landscape Perspective. *arXiv:1912.02757 [cs, stat]*, December 2019. arXiv: 1912.02757.

[9] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 369–376, Pittsburgh, Pennsylvania, USA, June 2006. Association for Computing Machinery.

[10] Dan Guest, Kyle Cranmer, and Daniel Whiteson. Deep Learning and its Application to LHC Physics. *Annual Review of Nuclear and Particle Science*, 68(1):161–181, October 2018. arXiv: 1806.11484.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015. arXiv: 1512.03385.

[12] Feifang Hu and James V. Zidek. The Weighted Likelihood. *The Canadian Journal of Statistics / La Revue Canadienne de Statistique*, 30(3):347–371, 2002.

[13] Nikos Karampatziakis and John Langford. Online importance weight aware updates. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, pages 392–399, Barcelona, Spain, July 2011. AUAI Press.

[14] Alex Kendall and Yarin Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5574–5584. Curran Associates, Inc., 2017.

[15] F. Kislat, B. Clark, M. Beilicke, and H. Krawczynski. Analyzing the data from X-ray polarimeters with Stokes parameters. *Astroparticle Physics*, 68:45–51, August 2015.

[16] Takao Kitaguchi, Kevin Black, Teruaki Enoto, Asami Hayato, Joanne E. Hill, Wataru B. Iwakiri, Philip Kaaret, Tsunefumi Mizuno, and Toru Tamagawa. A convolutional neural network approach for reconstructing polarization information of photoelectric X-ray polarimeters. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 942:162389, October 2019. arXiv: 1907.06442.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[18] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 6405–6416, Long Beach, California, USA, December 2017. Curran Associates Inc.

[19] Nikita Moriakov, Ashwin Samudre, Michela Negro, Fabian Gieseke, Sydney Otten, and Luc Hendriks. Inferring astrophysical X-ray polarization with deep learning. *arXiv e-prints*, 2005:arXiv:2005.08126, May 2020.

[20] Vikram Mullachery, Aniruddh Khera, and Amir Husain. Bayesian Neural Networks. *arXiv:1801.07710 [cs, stat]*, January 2018. arXiv: 1801.07710.

[21] J. A. Nelder and R. W. M. Wedderburn. Generalized Linear Models. *Journal of the Royal Statistical Society. Series A (General)*, 135(3):370–384, 1972. Publisher: [Royal Statistical Society, Wiley].

[22] George Papamakarios. Neural Density Estimation and Likelihood-free Inference. *arXiv:1910.13233 [cs, stat]*, October 2019. arXiv: 1910.13233.

[23] George Papamakarios, Theo Pavlakou, and Iain Murray. Masked Autoregressive Flow for Density Estimation. *arXiv:1705.07057 [cs, stat]*, June 2018. arXiv: 1705.07057.

[24] Tim Pearce, Mohamed Zaki, and Andy Neely. Bayesian Neural Network Ensembles. *arXiv:1811.12188 [cs, stat]*, November 2018. arXiv: 1811.12188.

[25] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[26] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection. *arXiv:1506.02640 [cs]*, June 2015. arXiv: 1506.02640.

[27] Lothar Rossol. Computer Vision in Industry. In Alan Pugh, editor, *Robot Vision*, International Trends in Manufacturing Technology, pages 11–18. Springer, Berlin, Heidelberg, 1983.

[28] Carmelo Sgro. The gas pixel detector on board the IXPE mission. In Oswald H. Siegmund, editor, *UV, X-Ray, and Gamma-Ray Space Instrumentation for Astronomy XX*, page 16, San Diego, United States, August 2017. SPIE.

[29] C. Sgrò and IXPE Team. The Imaging X-ray Polarimetry Explorer (IXPE). *Nuclear Instruments and Methods in Physics Research A*, 936:212–215, August 2019.

[30] Constantin Steppa and Tim Lukas Holch. HexagDLy - Processing hexagonally sampled data with CNNs in PyTorch. *SoftwareX*, 9:193–198, January 2019. arXiv: 1903.01814.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017.

[32] Martin Weisskopf. An Overview of X-Ray Polarimetry of Astronomical Sources. *Galaxies*, 6:33, March 2018.

[33] Martin C. Weisskopf, Ronald F. Elsner, and Stephen L. O'Dell. On understanding the figures of merit for detection and measurement of x-ray polarization. *arXiv:1006.3711 [astro-ph]*, page 77320E, July 2010. arXiv: 1006.3711.

[34] Andreas Wächter and Lorenz T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, March 2006.