PARAMETRIC DENSITY ESTIMATION WITH UNCERTAINTY USING DEEP ENSEMBLES

Anonymous authors
Paper under double-blind review

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 outperform current classical and deep learning methods.

1 INTRODUCTION

The majority of state-of-the-art NN performances are single (high-dimensional) input, multiple-output tasks, for instance classifying images (Krizhevsky et al., 2012), scene understanding (Redmon et al., 2015) and voice recognition (Graves et al., 2006). 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 that the data (or latent features of the data) are expected to 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.

One example of NNs making sense of large dataset-inputs is found in natural language processing. Here large text corpora, converted to word vectors (Pennington et al., 2014; Devlin et al., 2019), can be input and summarized by single output vectors using recurrent neural networks (RNNs), for instance in sentiment analysis (Can et al., 2018). However, these problems and RNNs themselves contain inductive bias – there is inherent structure in text. Not all information need be given at once and a concept of memory or attention is sufficient (Vaswani et al., 2017). The same can be said about time domain problems, such as audio processing or 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 direct PFDE computationally intractable.

On the other hand, density estimation on data directly (not on its latent features), is computationally tractable. Density estimation lets us find a complete statistical model of the data generating process. Applying deep learning to density estimation has advanced the field significantly (Papamakarios, 2019). 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 (Papamakarios

et al., 2018), or with parametric methods such as mixture density networks (Bishop, 1994). 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. A general solution is offered by deep ensembles (Lakshminarayanan et al., 2017). While these are not strictly equivalent to a Bayesian approach, although they can be made such using appropriate regularization (Pearce et al., 2018), they offer practical predictive uncertainties and have been shown to generalize readily (Fort et al., 2019).

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 bootstrap methods (Efron, 1979). 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

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[1] 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,

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

(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 (Lakshminarayanan et al., 2017) 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 aleatoric uncertainties. Final predictions and their epistemic uncertainties are then recovered by combining the estimates from each of the NNs in the ensemble.

In regression, deep ensembles model heteroscedastic aleatoric uncertainty by modifying the typical mean-squared errors (MSE) objective to a negative log-likelihood (NLL) Lakshminarayanan et al. (2017),

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

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 regression prediction and uncertainty are approximated by the mean and standard deviation over the M NN ensemble predictions respectively (each NN in the ensemble contributes equally). The epistemic uncertainty is then combined with the aleatoric in quadrature. Typically $M \sim 5 - 15$.

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 (Lakshminarayanan et al., 2017) (with recast loss function from Kendall & Gal (2017)) without using adversial examples, using the full dataset for each NN. Since the individual density parameters over predicted features are the final desired values in PFDE, it is possible that an objective maximizing feature accuracy on the validation set is not 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. This is especially true if the training dataset is biased. The single CNN method in fig. 2, §3.4, shows a clear case of training bias. If de-biasing the training dataset is not possible, we have identified two ways of resolving this issue for PDFE:

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.

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 Karampatziakis & Langford (2011). 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. Epistemic uncertainties are included in σ_{nm} by adding $\mathrm{Var}(\{y_{nm}\}_{m=1}^{M})$ in quadrature to each of the aleatoric uncertainty predictions. 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

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

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

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 Lakshminarayanan et al. (2017), 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. 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 Hu & Zidek (2002). 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) Nelder & Wedderburn (1972). 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 $f : \mathbb{R}^+ \rightarrow \mathbb{R}^+$ 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 bootstrap Efron (1979): 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 (Wächter & Biegler, 2006), 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 Feroz et al. (2009) 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 λ hyperparameter for the specific application.

3 EXPERIMENTS

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 Weisskopf (2018). The recent development of photoelectron tracking detectors (Bellazzini et al., 2003) 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. 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) (Sgrò & IXPE Team, 2019) 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 (Moriakov et al., 2020).

The linear polarization of light can be fully described by two degrees of freedom: the polarization fraction $0 \leq \Pi \leq 1$, (0% – 100%), and the electric vector position angle $-\pi/2 \leq \phi \leq \pi/2$. These can be thought of as the magnitude and direction of a vector perpendicular to the direction of propagation of the light. In imaging X-ray polarimetry, when the detector images an X-ray source, it measures individual 2D images of electron tracks excited by incoming X-ray photons. 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 the collected dataset of 2D track images.

In the case of IXPE, charge tracks are imaged by hexagonal pixels. 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 initial track angle θ follows the probability density

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

where Π and ϕ are fixed polarization parameters that depend on the source. 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. 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 (Sgro, 2017). 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 using a standard (unweighted) MLE. The moment analysis

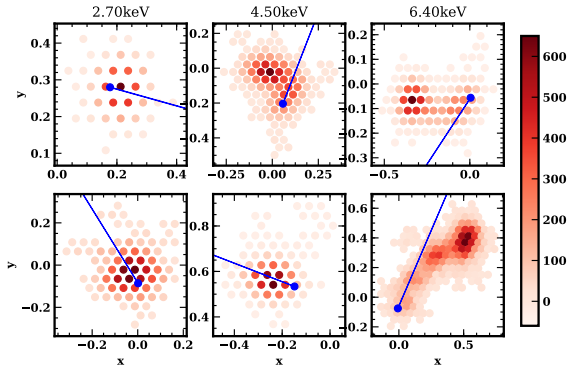


Figure 1: Example IXPE track images at 2.7, 4.5 and 6.4 keV energies (columns). The blue lines show the initial photoelectron direction. Color represents the amount of charge deposited in a hexagonal pixel. Track morphology (and thus angle reconstruction) depends strongly on energy.

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. NNs have also recently been applied to this problem Kitaguchi et al. (2019). 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 both the standard IXPE method and this basic NN approach.

3.2 Parametric feature density estimation

Following §2, we define CNNs that take single track images as input and $(\hat{\theta}, \hat{\sigma})$ 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 admissable inputs to standard 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 Steppa & Holch (2019). Since there are two possible shifts (odd and even rows), we apply both and stack the two shifted images, similar to color channels in rgb images. We do this to more closely approximate spatial equivariance of the CNN convolution kernels in the hexagonal space. At test time, we apply the deep ensemble to the same track 3 times, each time rotated by $120°$ in hexagonal space. We find this reduces all relevant prediction bias on $\hat{\theta}$ (and later Π, ϕ) introduced when converting from hexagonal to square coordinates.

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:

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

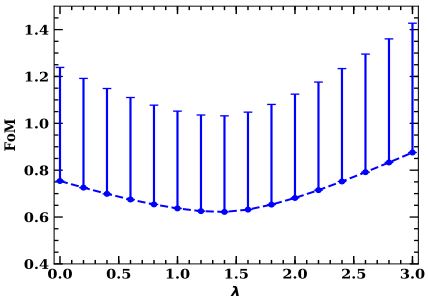
Each of the NNs in the ensemble output the 3-vector $(\hat{\mathbf{v}}, \hat{\sigma})$. Then $\hat{\theta} = \arctan\frac{\hat{\mathrm{v}}_2}{\hat{\mathrm{v}}_1}/2$. To calculate the final Π, ϕ with an ensemble of M NNs for a given test dataset with N tracks we minimize the importance weighted NLL (2)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

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

We can recast this as the convex optimization problem

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

where $\mathbf{v}_{nm} = (\cos\hat{\theta}_{nm}, \sin\hat{\theta}_{nm})$ 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.



Energy	**Moments**	**Mom. w/ cuts**	**Kitaguchi et al. (2019)**	**Our Method**	
	FoM (68% CI)	FoM	FoM	FoM	λ
2.7 keV	2.1 (2.7)	2.0 (2.6)	2.6 (3.3)	**1.7 (2.2)**	1.9
4.5 keV	1.3 (1.7)	1.2 (1.7)	1.5 (1.9)	**0.9 (1.3)**	1.1
6.4 keV	0.8 (1.2)	0.7 (1.1)	1.6 (1.9)	**0.5 (0.8)**	0.9
8.0 keV	0.7 (1.1)	0.7 (1.1)	0.8 (1.1)	**0.4 (0.6)**	0.8
PL 1	0.8 (1.2)	0.8 (1.3)	–	**0.5 (0.8)**	1.1
PL 2	1.0 (1.6)	0.9 (1.5)	–	**0.6 (1.0)**	1.3

Figure 2: *Right:* Results on energy selected track image datasets, comparing our method with the current state of the art. Lower FoM is better. PL1 and PL2 are power law datasets with range spanning $2.0 - 8.0$keV (PL1 $dN/dE \propto E^{-1}$, and PL2 $dN/dE \propto E^{-2}$). Individual energy datasets have 360 thousand tracks each, PL datasets have 1 million. All methods have $\text{RMSE}_\phi \leq 0.5°$. The weighting hyperparameter λ decreases with energy since the statistical error σ distribution is more closely clustered for lower energies. *Left:* FoM as a function of hyperparameter λ for the PL2 dataset. This method is used to select all of the λ. Confidence intervals (CI 68%) are calculated using the parametric bootstrap in both panels – note these are not the standard errors on FoM values, standard errors are $\text{CI}/\sqrt{10}$. For FoMs only the upper CI bound is necessary, since this represents the worst case signal to noise ratio.

3.2.1 FIGURE OF MERIT

In polarization estimation, we want high recovered $\hat{\Pi}_{100\%}$ (and accurate ϕ) for a known 100% polarized source ($\Pi_{100\%} = 1$), and low recovered $\hat{\Pi}_{0\%}$ for an unpolarized source ($\Pi_{0\%} = 0$). 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[2]: $\hat{\Pi} = \hat{\Pi}_{\text{meas}}/\hat{\Pi}_{100\%}$. We define a figure of merit for polarization estimation:

$$\text{FoM} = 100 \times \hat{\Pi}_{0\%}/\hat{\Pi}_{100\%}. \tag{8}$$

We use the FoM to evaluate model performance: a lower FoM means better polarization estimation. This is effectively a measure of the signal to noise ratio, a simplified extension of the minimum detectable polarization (MDP) typically defined for X-ray polarization (Weisskopf et al., 2010) that does not preclude biased estimators. It is evaluated on unseen polarized and unpolarized datasets. In estimating the FoM, we take the number of tracks $N \sim 360,000$ so we can compare directly to Kitaguchi et al. (2019). We average the FoM over 10 independent track dataset samples of size N. We use the FoM 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 training dataset consists of 3 million simulated tracks, examples of which are shown in fig. 1. The track energies uniformly span $1.0 - 9.0$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 use a ResNet-19 (He et al., 2015) 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. Before training we preprocess the training data (square track images). We apply pixelwise centering and rescaling. We use stochastic gradient descent with momentum and a decaying learning rate starting at $1e-2$. We choose batch sizes $512, 1024, 2048$ (tracks per batch). We trained for 70 epochs, saving NN checkpoints every 10 epochs for use in the NN selection. We use L_2-norm regularization 5×10^{-5}. Since NN θ predictions are unbiased §3.2, we choose $M = 10$ NNs with the top MSEs on θ for an unseen test dataset spanning all energies to make up our final NN ensemble.

[2] $\hat{\Pi}_{100\%}$ is measured before on a source with the same track energy distribution.

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 (Kitaguchi et al., 2019), provides significant improvements in $\Pi_{100\%}$ over the moment analysis, but adds bias to the unpolarized measurement, increasing its FoM 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 $\sim 2\times$ reduction in FoM. 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. We plan to release further results and more domain specific information for this particular application [*in prep., reference deleted to maintain integrity of review process*].

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 (Guest et al., 2018). Our approach is heuristically justified due to the asymptotic efficiency of the maximum likelihood estimator in a Cauchy location model (Cohen Freue, 2007). 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 (Rossol, 1983), 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 Bishop (1994) 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.

REFERENCES

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, pp. 383–393. International Society for Optics and Photonics, February 2003. doi: 10.1117/12.459381. URL https://www.spiedigitallibrary.org/conference-proceedings-of-spie/4843/0000/Novel-gaseous-x-ray-polarimeter-data-analysis-and-simulation/10.1117/12.459381.short.

Christopher Bishop. Mixture Density Networks. January 1994. URL https://www.microsoft.com/en-us/research/publication/mixture-density-networks/.

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. URL http://arxiv.org/abs/1806.04511. arXiv: 1806.04511.

Gabriela V. Cohen Freue. The Pitman estimator of the Cauchy location parameter. *Journal of Statistical Planning and Inference*, 137(6):1900–1913, June 2007. ISSN 0378-3758. doi: 10.1016/j.jspi.2006.05.002. URL http://www.sciencedirect.com/science/article/pii/S0378375806001285.

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. URL http://arxiv.org/abs/1810.04805. arXiv: 1810.04805.

B. Efron. Bootstrap Methods: Another Look at the Jackknife. *Annals of Statistics*, 7(1):1–26, January 1979. ISSN 0090-5364, 2168-8966. doi: 10.1214/aos/1176344552. URL https://projecteuclid.org/euclid.aos/1176344552. Publisher: Institute of Mathematical Statistics.

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. ISSN 00358711, 13652966. doi: 10.1111/j.1365-2966.2009.14548.x. URL http://arxiv.org/abs/0809.3437. arXiv: 0809.3437.

Stanislav Fort, Huiyi Hu, and Balaji Lakshminarayanan. Deep Ensembles: A Loss Landscape Perspective. *arXiv:1912.02757 [cs, stat]*, December 2019. URL http://arxiv.org/abs/1912.02757. arXiv: 1912.02757.

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, pp. 369–376, Pittsburgh, Pennsylvania, USA, June 2006. Association for Computing Machinery. ISBN 978-1-59593-383-6. doi: 10.1145/1143844.1143891. URL https://doi.org/10.1145/1143844.1143891.

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. ISSN 0163-8998, 1545-4134. doi: 10.1146/annurev-nucl-101917-021019. URL http://arxiv.org/abs/1806.11484. arXiv: 1806.11484.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385 [cs]*, December 2015. URL http://arxiv.org/abs/1512.03385. arXiv: 1512.03385.

Feifang Hu and James V. Zidek. The Weighted Likelihood. *The Canadian Journal of Statistics / La Revue Canadienne de Statistique*, 30(3):347–371, 2002. ISSN 0319-5724. doi: 10.2307/3316141. URL https://www.jstor.org/stable/3316141.

Nikos Karampatziakis and John Langford. Online importance weight aware updates. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, UAI'11, pp. 392–399, Barcelona, Spain, July 2011. AUAI Press. ISBN 978-0-9749039-7-2.

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 (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5574–5584. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7141-what-uncertainties-do-we-need-in-bayesian-deep-learning-for-computer-vision.pdf.

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. ISSN 01689002. doi: 10.1016/j.nima.2019.162389. URL http://arxiv.org/abs/1907.06442. arXiv: 1907.06442.

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 (eds.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

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, pp. 6405–6416, Long Beach, California, USA, December 2017. Curran Associates Inc. ISBN 978-1-5108-6096-4.

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. URL http://adsabs.harvard.edu/abs/2020arXiv200508126M.

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. ISSN 0035-9238. doi: 10.2307/2344614. URL https://www.jstor.org/stable/2344614. Publisher: [Royal Statistical Society, Wiley].

George Papamakarios. Neural Density Estimation and Likelihood-free Inference. *arXiv:1910.13233 [cs, stat]*, October 2019. URL http://arxiv.org/abs/1910.13233. arXiv: 1910.13233.

George Papamakarios, Theo Pavlakou, and Iain Murray. Masked Autoregressive Flow for Density Estimation. *arXiv:1705.07057 [cs, stat]*, June 2018. URL http://arxiv.org/abs/1705.07057. arXiv: 1705.07057.

Tim Pearce, Mohamed Zaki, and Andy Neely. Bayesian Neural Network Ensembles. *arXiv:1811.12188 [cs, stat]*, November 2018. URL http://arxiv.org/abs/1811.12188. arXiv: 1811.12188.

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)*, pp. 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://www.aclweb.org/anthology/D14-1162.

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

Lothar Rossol. Computer Vision in Industry. In Alan Pugh (ed.), *Robot Vision*, International Trends in Manufacturing Technology, pp. 11–18. Springer, Berlin, Heidelberg, 1983. ISBN 978-3-662-09771-7. doi: 10.1007/978-3-662-09771-7_2. URL https://doi.org/10.1007/978-3-662-09771-7_2.

Carmelo Sgro. The gas pixel detector on board the IXPE mission. In Oswald H. Siegmund (ed.), *UV, X-Ray, and Gamma-Ray Space Instrumentation for Astronomy XX*, pp. 16, San Diego, United States, August 2017. SPIE. ISBN 978-1-5106-1251-8 978-1-5106-1252-5. doi: 10.1117/12.2273922. URL https://www.spiedigitallibrary.org/conference-proceedings-of-spie/10397/2273922/The-gas-pixel-detector-on-board-the-IXPE-mission/10.1117/12.2273922.full.

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. ISSN 0168-9002. doi: 10.1016/j.nima.2018.10.111. URL http://adsabs.harvard.edu/abs/2019NIMPA.936..212S.

Constantin Steppa and Tim Lukas Holch. HexagDLy - Processing hexagonally sampled data with CNNs in PyTorch. *SoftwareX*, 9:193–198, January 2019. ISSN 23527110. doi: 10.1016/j.softx.2019.02.010. URL http://arxiv.org/abs/1903.01814. arXiv: 1903.01814.

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 (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Martin Weisskopf. An Overview of X-Ray Polarimetry of Astronomical Sources. *Galaxies*, 6:33, March 2018. doi: 10.3390/galaxies6010033. URL http://adsabs.harvard.edu/abs/2018Galax...6...33W.

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]*, pp. 77320E, July 2010. doi: 10.1117/12.857357. URL http://arxiv.org/abs/1006.3711. arXiv: 1006.3711.

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. ISSN 1436-4646. doi: 10.1007/s10107-004-0559-y. URL https://doi.org/10.1007/s10107-004-0559-y.