
Proceedings Paper:
Carannante, G., Bouaynaya, N.C. and Mihaylova, L. orcid.org/0000-0001-5856-2223 (2021) An enhanced particle filter for uncertainty quantification in neural networks. In: de Villiers, P., de Waal, A. and Gustafsson, F., (eds.) 2021 IEEE 24th International Conference on Information Fusion (FUSION). 24th International Conference on Information Fusion (Fusion 2021), 01-04 Nov 2021, Sun City, South Africa. Institute of Electrical and Electronics Engineers . ISBN 9781665414272

An Enhanced Particle Filter for Uncertainty Quantification in Neural Networks

Giuseppina Carannante*, Nidhal C. Bouaynaya† and Lyudmila Mihaylova‡

*†*Department of Electrical and Computer Engineering*, Rowan University, Glassboro, New Jersey, USA
‡*Department of Automatic Control and Systems Engineering*, The University of Sheffield, Sheffield, United Kingdom
Email: *carannang1@rowan.edu, †bouaynaya@rowan.edu, ‡l.s.mihaylova@sheffield.ac.uk

***Abstract*—The brittleness of deep learning models is ailing their deployment in real-world applications, such as transportation and airport security. Most work focuses on developing accurate models that only deliver point estimates without further information on model uncertainty or confidence. Ideally, a learning model should compute the posterior predictive distribution, which contains all information about the model output. We cast the problem of density tracking in neural networks using Particle Filtering, a powerful class of numerical methods for the solution of optimal estimation problems in non-linear, non-Gaussian systems. Particle filters are a powerful alternative to Markov chain Monte Carlo algorithms and enjoy established convergence and performance guarantees. In this paper, we advance a particle filtering framework for neural networks, where the predictive output is a distribution. The mean of this distribution serves as the point estimate decision and its variance provides the model confidence in the decision. Our framework shows increased robustness under noisy conditions. Additionally, the predictive variance increases monotonically with decreasing signal-to-noise ratio (SNR); thus reflecting a lower confidence or higher uncertainty. This paper serves as a pioneering proof-of-concept framework that will allow the development of a theoretical understanding of robust neural networks.**

***Index Terms*—Bayesian Learning, Particle Filtering, Neural Networks, Uncertainty Quantification.**

I. Introduction

Bayesian approaches in neural networks are being revisited due to their ability to provide much needed uncertainty information, which is not carried by classical "point estimate" machine learning approaches, such as neural networks. The inability to evaluate a system's uncertainty or confidence in its output can have disastrous consequences, especially when the output of the model is fed into higher-level decision making processes. Such models include recommendation systems in the medical domain, autonomous drones and vehicles, and high frequency trading [1]–[3]. In fact, model uncertainty information is a requirement for successful deployment of the system. Unlike deterministic learning methods that aim at estimating a single set of parameters (weights and biases), the Bayesian setting poses a distribution over the network parameters [4]. Through Bayesian inference, the posterior predictive distribution can be derived and the variance of this distribution is then used as a measure of model confidence.

However, exact Bayesian inference in neural networks is mathematically intractable as it involves propagation of distributions across non-linearities. Consequently, approximation or numerical techniques are needed. Markov Chain Monte Carlo (MCMC) methods were proposed jointly with neural networks to tackle Bayesian inferential and prediction problems. The main challenges posed by neural networks to MCMC developments include lack of parameter identifiability due to weight symmetries, prior specification effects, and high computational cost [5]. Variants to MCMC were proposed to mitigate these issues, but these methods do not deliver the same convergence guarantees as the basic MCMC [6], [7]. The development and applicability of these approaches remains limited.

On the other hand, approaches like Monte Carlo (MC) - Dropout [8] that implicitly perform Bayesian inference, have gained great success [9]–[11]. These methods exploit regularization techniques and/or injection of noise as means to produce samples from an approximate distribution, e.g. Bernoulli. Considering an ensemble of models allows to compute the predictive uncertainty at inference time by evaluating the variation of the predictions. Even though these methods are very simple, since their integration into current neural network training is straightforward, they still suffer from the demand of sampling at inference time. Additionally, ensemble models are not considered as fully Bayesian since they rely on a deterministic training.

A popular approximation method is based on the Variational Inference (VI) technique [12]. In VI, the inference is formulated as an optimization problem, where the Kullback-Leibler (KL) divergence between the posterior distribution and an approximate parametric distribution is minimized. Several scalable approaches have been developed within the VI framework, including some of our own work [13]–[16]. However, this optimization process relies on a known form of the approximating distribution, thus constraining the posterior density to be of a particular shape. For example, Bayes by BackProp (BBB) [14] places a fully-factorized Gaussian over the network parameters and samples one set of parameters from this approximating distribution for each forward pass. Consequently, the predictive power of these approaches is still limited by the Gaussian assumption.

This work is inspired by powerful statistical frameworks for optimal tracking in non-linear and non-Gaussian dynamical systems. Different from the VI framework, we do not impose

a parametric form on the posterior predictive distribution and develop a Particle Filter (PF) solution to track this posterior across the network non-linearities. The contribution of this paper can be summarized as follows:

- We develop a novel framework to track the posterior distribution through the neural network's layers without imposing any parametric constraint on the form of such distribution.
- The posterior is tracked using a set of weighted *particles*, which can be used to estimate moments of any order. In particular, these samples, with their corresponding weights, are employed to compute the second moment, i.e., the predictive variance and render the model's confidence.
- We study the second moment behaviour under noisy conditions. We show that the predictive variance monotonically increases as the Signal-to-Noise Ratio (SNR) decreases, reflecting a higher uncertainty or lower confidence in the model prediction.

The paper is organized as follows: in Section II, we review the fundamentals of Bayesian learning and particle filtering. The proposed enhanced particle filter (E-PF) for neural networks is presented in Section III. Our experiments and a discussion of the results are provided in Section IV. Conclusion remarks and future work are given in Section V.

II. Bayesian Learning and Particle Filtering

For the completeness of the paper, we first provide a brief background review of Bayesian learning, state-space modeling and Particle Filtering.

A. Bayesian Learning

In Bayesian neural networks (BNNs), the parameters $\mathcal{W}$ are interpreted as random variables with a prior distribution, i.e., $\mathcal{W} \sim p(\mathcal{W})$. As data $\mathcal{D}$ is observed, we compute the likelihood $p(\mathcal{D}|\mathcal{W})$ and infer the posterior probability density function $p(\mathcal{W}|\mathcal{D})$ using Bayes' Theorem. By inferring the posterior, we are able to compute the predictive distribution, i.e., the distribution of unseen data points:

$$p(\tilde{\mathbf{y}}|\tilde{\mathbf{x}}, \mathcal{D}) = \int p(\tilde{\mathbf{y}}|\tilde{\mathbf{x}}, \mathcal{W})\ p(\mathcal{W}|\mathcal{D})\ d\mathcal{W}, \tag{1}$$

where $\tilde{\mathbf{x}}$ and $\tilde{\mathbf{y}}$ denote, respectively, the unseen input and its corresponding output. The predictive distribution carries all information about the prediction; in particular, the mean represents the network prediction while the variance is interpreted as confidence (or uncertainty) information.

Various approaches have been proposed to estimate predictions' uncertainty. For a comprehensive review of popular and most recent methods for uncertainty quantification in machine learning we refer the reader to [17].

B. State-Space Models

The general state-space model formulation for a dynamical system is given by

$$\begin{aligned} \boldsymbol{\theta}_{k+1} &= G(\mathbf{X}, \boldsymbol{\theta}_k), \\ \boldsymbol{y}_k &= F(\mathbf{X}, \boldsymbol{\theta}_k), \end{aligned} \tag{2}$$

where $\boldsymbol{\theta}_k$ represents the hidden state at time k, $\boldsymbol{X}$ is the input, and $\boldsymbol{y}_k$ is the output vector. The potentially non-linear maps G and F represent, respectively, the state-transition and measurement models that are nonlinear in general.

Training multilayer perceptrons was formulated as an identification problem for a dynamic system modeled with a state-space representation, which can be solved using the Extended Kalman filter and sequential Monte Carlo methods [18]–[22]. This rendering is mathematically grounded given that parameters of a neural network (NN) can be seen as a discrete-time system that evolves as data is seen.

Within the neural network framework, the state $\boldsymbol{\theta}_k$ represents the vector of parameters of the NN at *epoch* k. The map G describes the evolution of the parameters during training while the measurement function F is approximated via the NN input-output mapping. In a stochastic framework, $\boldsymbol{\theta}_k$ and $\boldsymbol{y}_k$ are treated as random variables, and the filtering problem amounts to finding the posterior density $p(\boldsymbol{\theta}_k|\boldsymbol{Y}_{1:k})$, where $\boldsymbol{Y}_{1:k} = \{\boldsymbol{y}_i\}_{i=1}^{k}$ denotes the history of observations. This filtering distribution is obtained sequentially by alternating an update step for the hidden state followed by a measurement evaluation step.

C. Particle Filtering

Particle Filtering (PF) is a flexible and powerful sequential MC method designed to solve numerically the optimal filtering problem within non-linear and non-Gaussian systems [23]–[26]. Moreover, PF converges asymptotically toward the optimal filter in the mean square error sense [27]. PF employs a set of weighted samples, also called *particles*, to approximate the posterior distribution. In particular, the posterior of the state is approximated by a large set of Dirac-delta masses (samples/particles) that evolve stochastically in time according to the dynamics of the model and the observations.

PF relies on the fundamental method of *Importance Sampling* (IS) which builds on the introduction of an *importance distribution*, $\pi(\boldsymbol{\theta})$. Unlike the *true* posterior, $p(\boldsymbol{\theta}|\boldsymbol{Y})$, the *importance distribution* has a simpler form so it is easy to draw samples from, e.g., a multivariate Gaussian.

To adjust for the posterior distribution, an appropriate weight is computed for each particle based on the available observations, as follows:

$$\boldsymbol{\theta}^{(i)} \sim \pi(\boldsymbol{\theta}), \quad u^{(i)} = \frac{p(\boldsymbol{\theta}^{(i)})}{\pi(\boldsymbol{\theta}^{(i)})} \qquad \text{for} \quad i = 1, 2, \ldots N. \tag{3}$$

Hence, each importance weight, $u^{(i)}$, reflects the likelihood of the particle (sampled from the importance distribution) to be a probable realization of the *true* posterior.

The set of weighted particles is used to form a numerical approximation for the posterior. Additionally, the PF allows for

Algorithm 1 E-PF

for $k = 1, 2, \ldots, K$ **do**
  for $l = 1, 2, \ldots, L$ **do**
    for $i = 1, 2, \ldots, N$ **do**
      $\eta^{(i)} \sim \mathcal{N}(0,\, \sigma_\eta^2 \mathbf{I})$
      $\mathbf{W}_{[l][k]}^{(i)} \leftarrow \mathbf{W}_{[l][k-1]} + \eta^{(i)}$
      Evaluate $\boldsymbol{y}^{(i)} = F(\mathbf{X}, \mathbf{W}_{[1][k]}, \ldots, \mathbf{W}_{[l-1][k]}, \mathbf{W}_{[l][k]}^{(i)}, \mathbf{W}_{[l+1][k-1]}, \ldots, \mathbf{W}_{[L][k-1]})$
      Evaluate the quality of fit of the particle through $\mathcal{L}(\boldsymbol{y}^{(i)})$
      Compute $u_{[l]}^{(i)} \propto \mathcal{L}(\boldsymbol{y}^{(i)})$
    end for
    Normalize importance ratios $\widetilde{u}_{[l]}^{(i)} = \frac{u_{[l]}^{(i)}}{\sum_j u_{[l]}^{(j)}}$ for $i = 1, 2, \ldots, N$
    Resample $\mathbf{W}_{[l][k]}^{(i)}$ and associate a weight of $1/N$ with each offspring
    Estimate $\mathbf{W}_{[l][k]} \leftarrow \frac{1}{N} \sum_i \mathbf{W}_{[l][k]}^{(i)}$
  end for
  perform a forward pass using $\mathcal{W}_k$: $\boldsymbol{y} = F(\mathbf{X}, \mathcal{W}_k)$
  Compute loss $\mathcal{L}(\boldsymbol{y})$
  Update $\mathcal{W}_k \leftarrow \mathcal{W}_k - \alpha \nabla_{\mathcal{W}} \mathcal{L}$
end for

output: $\mathbf{W}_{[l][K]}^{(i)}, \widetilde{u}_{[l]}^{(i)}, \quad i = 1, 2, \ldots, N \quad \text{and} \quad l = 1, 2, \ldots, L$

the evaluation of any mathematical expectation of the form:

$$\mathbb{E}_{\boldsymbol{\theta} \sim p(\boldsymbol{\theta})}(h(\boldsymbol{\theta})) = \int_{\boldsymbol{\theta}} h(\boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta} = \int_{\boldsymbol{\theta}} h(\boldsymbol{\theta}) \frac{p(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta})} \pi(\boldsymbol{\theta}) d\boldsymbol{\theta}$$
$$= \mathbb{E}_{\boldsymbol{\theta} \sim \pi(\boldsymbol{\theta})} \left[h(\boldsymbol{\theta}) \frac{p(\boldsymbol{\theta})}{\pi(\boldsymbol{\theta})} \right] \approx \frac{1}{N} \sum_{i=1}^{N} u^{(i)} h(\boldsymbol{\theta}^{(i)}), \tag{4}$$

where h is any function and we have simplified the notation for the *true* posterior, i.e., $p(\boldsymbol{\theta}) := p(\boldsymbol{\theta}|\boldsymbol{Y})$. The notation $\mathbb{E}$ refers to the mathematical expectation.

In sequential estimation, the importance weights are computed as new evidence data is acquired based on the measurement model and the posterior is updated accordingly. In PF, the variance of the importance weights increases over time. To mitigate this issue, resampling schemes have been proposed in the literature. For detailed derivations, importance density choices, resampling schemes and modifications of PF methods, the reader is referred to surveys and tutorials [28]–[30].

III. Enhanced Particle Filter for Neural Networks

Throughout the rest of this paper, we refer to the available data as $\mathcal{D} = \{(\mathbf{x}, \mathbf{y})_j\}_{j=1}^{M}$, M being the total number of datapoints. $\mathcal{W} = \{\mathbf{W}_{[1]}, \ldots, \mathbf{W}_{[L]}\}$ is the set of network parameters, where L is number of layers and $\mathbf{W}_{[l]}$ incorporates both the weights and biases for layer l. $\mathcal{W}_k$ denotes all network parameters (in all layers) estimated at time-step k, i.e., $\mathcal{W}_k = \{\mathbf{W}_{[l][k]}\}_{l=1}^{L}$. The notation $\mathbf{W}_{[l][k]}^{(i)}$ is employed to denote the i^{th} particle of layer l at time-step k. Finally, to avoid any confusion, we refer to the *weights* $u^{(i)}$ of the PF process as *importance ratios* to circumvent the potential misinterpretation with the weights of the neural network. Additionally, we employ the notation $u_{[l]}^{(i)}$ to specify the layer l.

We formulate the learning problem using a state-space model as in (2). The system dynamic equation G is chosen as the Gradient Descend (GD) update rule, i.e.,

$$\begin{aligned} \mathcal{W}_{k+1} &= \mathcal{W}_k - \alpha \nabla_{\mathcal{W}} \mathcal{L}, \\ \boldsymbol{y}_k &= F(\mathbf{X}, \mathcal{W}_k), \end{aligned} \tag{5}$$

where $\mathcal{L}$ is the loss function for the network and α is the learning rate.

This idea of an *enhanced* transition model has been previously applied in the context of NN training for pricing option contracts traded in financial markets [22]. It was shown that this choice of parameter evolution leads to a better convergence as compared to a random-walk model [22]. However, unlike the previous and other methods that exploit the state-space framework, we break the state vector into L components, one for each layer's parameters $\mathbf{W}_{[l]}$, and perform PF separately for each of these. We elaborate this trick to mitigate the high-dimensionality problem of PF. In particular, we perform L forward passes through the network to evaluate the map F and compute the loss $\mathcal{L}$ for each particle.

In detail, at time-step $k = 0$, for each layer l, we initialize $\mathbf{W}_{[l]}$ drawing from a prior distribution $p(\mathbf{W}_{[l]})$. If we let N denote the number of particles and K the total number of

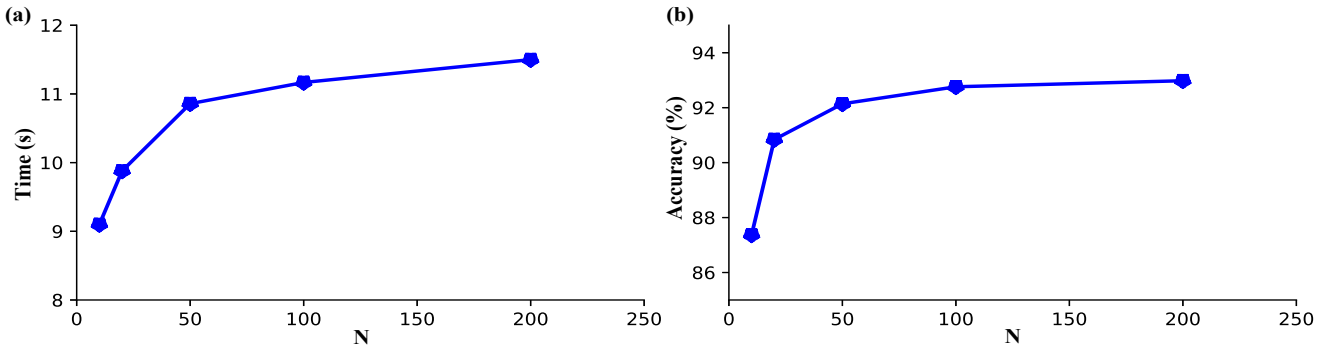

Fig. 1. We trained and tested our E-PF for various numbers of particles N. The reported results are obtained after training for one time-step (epoch). We plot the training time vs the number of particles N in (a) while we report the testing accuracy in (b). As expected, both time and accuracy are increasing as N increases.

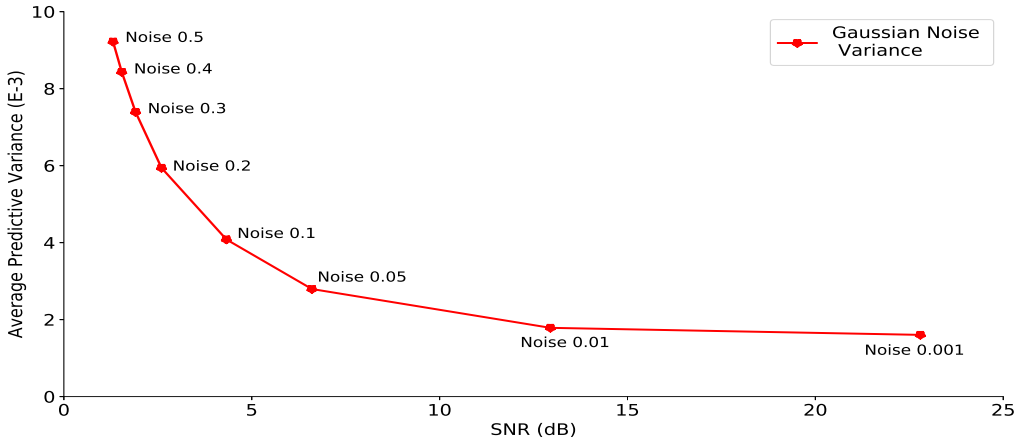

Fig. 2. The E-PF Predictive Variance at various levels of signal-to-noise ratio (SNR) from additive Gaussian Noise. The Predictive variance is defined as the variance value corresponding to the label predicted by the network. The reported values are averaged over all images in the test set.

time-steps (epochs), the steps of the proposed E-PF algorithm are provided above in algorithm 1.

In the NN set up, we assess the goodness of fit of each particle through the loss function. Specifically, we want to favor particles with lower loss values. Hence, we set the importance ratios $u^{(i)}$ to be proportional to the loss $\mathcal{L}$ of the network, e.g., cross-entropy loss in classification problems. In our implementation, the importance ratios for each layer l are computed using the most recent estimates $\mathbf{W}_{[j][k]}$ for layers $j = 1, 2, \ldots, l-1$ and the previous time-step estimates, i.e., $\mathbf{W}_{[j][k-1]}$, for layers $j = l+1, l+2, \ldots, L$. At the end of the training, particles and normalized importance ratios,

$$\{\mathbf{W}^{(i)}_{[l][K]}, \widetilde{u}^{(i)}_{[l]} = \frac{u^{(i)}_{[l]}}{\sum_j u^{(j)}_{[l]}}\}_{i=1}^{N},$$

are used to approximate the posterior distribution of $\mathbf{W}_{[l]}$, for $l = 1, 2, \ldots, L$.

As mentioned in Section II-A, once the posterior is approximated, we are able to derive the predictive distribution. In addition, the availability of weighted particles enables us to compute any order moment. The second moment of the predictive distribution, i.e., the predictive variance, is of particular interest as a measure of model uncertainty or confidence. Specifically, we can employ the dispersion of the particles cloud as an approximation for the variance.

IV. Performance Validation and Evaluation

We evaluate our approach, E-PF, on a classification task using MNIST handwritten digits dataset [31]. We compare the performance of the proposed approach with state-of-the-art frameworks. We consider two popular Bayesian approaches MC-Dropout and BBB [8], [14]. For our framework, we conduct a noise analysis and study the predictive variance information.

A. Experiments

We evaluate the performance of E-PF using a two-layers FC NN with 50 and 10 neurons, respectively, followed by ReLU and Softmax activations. We employ cross-entropy loss with L_2 regularization. For the MNIST data, the number of classes is $C = 10$. We use the classical split of $60,000$ images for training and $10,000$ for testing. We employ a

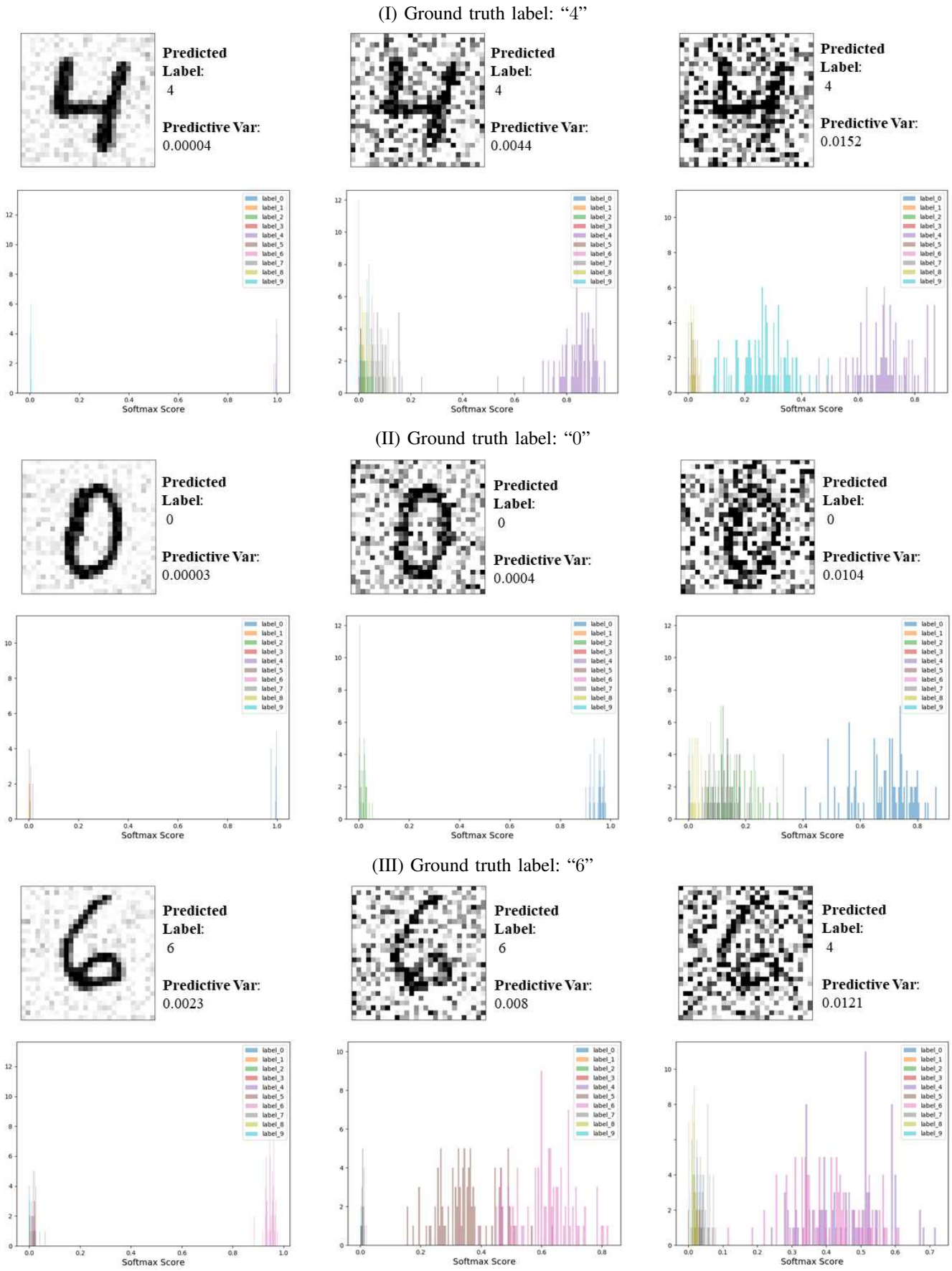

Fig. 3. Predictive distributions of E-PF for three randomly chosen images from the MNIST test dataset with added Gaussian noise with variance equal to 0.01, 0.2 and 0.4 (left to right). For each test image, the top row shows the noisy images along with the E-PF predicted labels, i.e., the labels corresponding to the mean predictions, and the predictive variance values. The bottom row shows the particles' predictions (particles softmax scores). Observe that the variances of the predictive distributions increase with increasing noise levels, reflecting higher uncertainty (lower confidence) into the prediction.

Gaussian distribution as the prior for both layers: $\mathcal{N}(0, \sigma^2\mathbf{I})$ with $\sigma = 0.1$. For our E-PF, we consider $N = 100$ particles. To generate these particles (see alg. 1), we set the standard deviation value σ_η to a constant value of 0.05 for both layers. To weight the particles, we compute the importance ratios as $-\sum_i^C t_i log(p_i)$, i.e., as the cross-entropy loss computed comparing the true label t_i and the particle's predicted output p_i. For this experiment we compare our E-PF with MC-Dropout and BBB. At inference time, we perform and average 100 forward passes predictions for both MC-Dropout and BBB. We show the performance of the three approaches on noise-free data and at various levels of Gaussian noise (see Table I). While Table I reports accuracy results for $N = 100$ particles, Fig. 1 exhibits the performance and computational cost of E-PF when we make N vary. In Fig. 2, we show the behaviour of the predictive variance of E-PF when several levels of Gaussian noise are added to the test data. In particular, Fig. 3 displays the predictive distributions for three randomly selected images from the test set when Gaussian noise is added.

TABLE I
TEST ACCURACY FOR E-PF, BAYES BY BACKPROP, AND MC-DROPOUT

Noise Level	E-PF	BBB	MC-Dropout
Zero	96%	96%	96%
Low (Var = 0.01)	95%	95%	94%
Medium (Var = 0.2)	75%	74%	70%
High (Var = 0.4)	47%	46%	40%

B. Discussion

Table I shows that all Bayesian approaches have comparable accuracy on noise-free data. BBB and the proposed E-PF NN outperform MC-dropout on noisy data with E-PF's accuracy slightly higher than BBB. Beyond accuracy, E-PF provides valuable information about uncertainty quantification, i.e., the predictive variance. We define such quantity as the variance corresponding to the label predicted by the network. Table II reports the predictive variance for the noise-free case. It is known that most models produce overconfident predictions [32], [33]. Note that the predictive variance value is larger for incorrect predictions as compared to the average value for all test images and the value for correctly classified images (the smallest). Hence, our model is able to highlight *what it knows and what it does not know*.

A study of the model response when a Gaussian noise is added to the test data has been performed. The focus of this paper is on the first and second order moments, i.e., the predicted mean and the predicted variance, but the full predictive distribution is available and any order statistic can be computed and examined under noisy conditions. Figure 2 shows the predicted variance versus the Signal-to-Noise Ratio (SNR). It is very interesting to observe that the variance increases monotonically with decreasing SNR values. For higher values of added Gaussian noise, the network becomes more uncertain which is reflected in an increasing predictive variance.

TABLE II
E-PF PREDICTIVE VARIANCE

	All	Correct	Incorrect
Predictive variance	0.0016	0.0012	0.0098

This response to noise is also demonstrated in Fig. 3 which displays the predictive distribution of E-PF for three randomly selected images from the test set. Given the classification task, it is informative to display the particles' softmax score for each label. For each test image, we show the noisy image and report the network predicted label, i.e., label corresponding to the mean prediction, and the predictive variance value. For lower noise levels, the network is highly confident about the predictions: the distribution is peaked around the mean with a very small spread (low uncertainty). As noise increases, the spread of the particles' prediction increases (higher uncertainty) with a more noticeable skew which suggests that some of the particles are moving toward an incorrect prediction. This is evident in Fig. 3 (III - noise level variance $= 0.4$): the intersection shows that some particles are correctly classifying label 6 but others are incorrectly assigning 4 as the predicted label. Observe that, although the network is assigning the incorrect class label, the predictive variance is 50% higher, reflecting a significant decrease in confidence.

We are proposing a shift in evaluating neural networks' performance with a dual metric: (accuracy, variance). The E-PF approach's predictive variance can be used as a *warning sign* since higher values indicate noisy conditions and/or possibly incorrect predictions. Additionally, the full predictive distribution can be plotted and used as an additional valuable tool to assess the prediction's reliability.

It is worth to mention that the PF guarantees under weak assumption convergence to the *true* posterior. In particular, if we denote with N the number of particles in the E-PF algorithm, as we let $N \to \infty$, the obtained E-PF approximation will be preferred. However, as in many other ensemble approaches, as we increase the particles' cloud (number of samples N), the computational demand of the algorithm will grow. As we can note from Fig. 1, both training time and test accuracy increase as we increase the number of particles N.

V. CONCLUSIONS

This paper proposes a Particle-Filter approach that propagates densities through the non-linear layers of a neural network. We leveraged fundamental concepts from powerful statistical frameworks in optimal estimation problems in non-linear non-Gaussian systems. By propagating a set of random samples through the layers of a neural network, we can approximate the posterior distribution without relying on crude functional approximations. Furthermore, particle filtering guarantees, under weak assumptions, asymptotic convergence (i.e., when $N \to \infty$) and consistent estimates of the posterior

distribution. We exploit the PF framework to derive the predictive variance and study its behaviour under noisy conditions. We observed that the variance of the predictive distribution increases monotonically with decreasing SNR values, which reflects a higher uncertainty or lower confidence into the decision made by the network. The results of this paper, although preliminary, open the door to a further investigation into the issues of robustness and trustworthiness of neural networks. Future work includes scaling the proposed E-PF NN to convolutional neural networks and sequence models, exploring the role of higher moments of the predictive distribution and going beyond Gaussian noise, i.e., adversarial robustness.

ACKNOWLEDGEMENT

This work was supported by The US National Science Foundation (NSF) Award ECCS-1903466 and the UK EPSRC [Grant No. EP/ T013265/1, NSF-EPSRC: "ShiRAS. Towards Safe and Reliable Autonomy in Sensor Driven"] and the UK Research and Innovation (UKRI) Trustworthy Autonomous Systems (TAS) programme [EPSRC Ref: EP/ V026747/1]. Giuseppina Carannante is supported by the US Department of Education through a Graduate Assistance in Areas of National Need (GAANN) program Award Number P200A180055.

REFERENCES

[1] E. Ackerman, "How drive.ai is Mastering Autonomous Driving with Deep Dearning," *IEEE Spectrum Magazine*, Mar. 2017.

[2] J. Ker, L. Wang, J. Rao, and T. Lim, "Deep Learning Applications in Medical Image Analysis," *IEEE Access*, vol. 6, pp. 9375–9389, 2017.

[3] C. M. Bishop, "Novelty Detection and Neural Network Validation," *IEE Proceedings-Vision, Image and Signal processing*, vol. 141, no. 4, pp. 217–222, 1994.

[4] D. J. MacKay, "A Practical Bayesian Framework for Backpropagation Networks," *Neural Computation*, vol. 4, no. 3, pp. 448–472, 1992.

[5] R. M. Neal, *Bayesian Learning for Neural Networks*, vol. 118, Springer Science & Business Media, 2012.

[6] M. Welling and Y. W. Teh, "Bayesian Learning via Stochastic Gradient Langevin Dynamics," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, 2011, pp. 681–688.

[7] J. T. Springenberg, A. Klein, S. Falkner, and F. Hutter, "Bayesian Optimization with Robust Bayesian Neural Networks," in *Advances in neural information processing systems*, 2016, pp. 4134–4142.

[8] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *Proc. of International Conference on Machine Learning*, 2016, pp. 1050–1059.

[9] D. P. Kingma, T. Salimans, and M. Welling, "Variational Dropout and the Local Reparameterization Trick," in *Advances in Neural Information Processing Systems*, 2015, pp. 2575–2583.

[10] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles," in *Advances in Neural Information Processing Systems*, 2017, pp. 6402–6413.

[11] X. Lu and B. Van Roy, "Ensemble Sampling," in *Advances in neural information processing systems*, 2017, pp. 3258–3266.

[12] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An Introduction to Variational Methods for Graphical Models," in *Learning in Graphical Models*, pp. 105–161. Springer, 1998.

[13] A. Graves, "Practical Variational Inference for Neural Networks," in *Advances in Neural Information Processing Systems*, 2011, pp. 2348–2356.

[14] C. Blundell, J. Cornebise, K. Kavukcuoglu D., and Wierstra, "Weight Uncertainty in Neural Networks," *arXiv preprint arXiv:1505.05424*, 2015.

[15] D. Dera, G. Rasool, and N. C. Bouaynaya, "Extended Variational Inference for Propagating Uncertainty in Convolutional Neural Networks," in *Proc. of the IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*. IEEE, 2019, pp. 1–6.

[16] G. Carannante, D. Dera, G. Rasool, N. C. Bouaynaya, and L. S. Mihaylova, "Robust Learning via Ensemble Density Propagation in Deep Neural Networks," in *Proc. of the IEEE International Workshop on Machine Learning for Signal Processing*. IEEE, 2020.

[17] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, V. Makarenkov, and S. Nahavandi, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *arXiv preprint arXiv:2011.06225*, 2020.

[18] S. Singhal and L. Wu, "Training Multilayer Perceptrons with the Extended Kalman Algorithm," in *Advances in Neural Information Processing Systems*, 1989, pp. 133–140.

[19] G. V. Puskorius and L. A. Feldkamp, "Decoupled Extended Kalman Filter Training of Feedforward Layered Networks," in *Proc. of the IJCNN-91-Seattle International Joint Conference on Neural Networks*. IEEE, 1991, vol. 1, pp. 771–777.

[20] J. F. G. Freitas, S. E. Johnson, M. Niranjan, and A. H. Gee, "Global Optimisation of Neural Network Models via Sequential Sampling-Smportance Resampling," in *Proc. of the Fifth International Conference on Spoken Language Processing*, 1998.

[21] E. A. Wan and R. Van Der Merwe, "The Unscented Kalman Filter for Nonlinear Estimation," in *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373)*. IEEE, 2000, pp. 153–158.

[22] J. F. G. Freitas, M. Niranjan, A. H. Gee, and A. Doucet, "Sequential Monte Carlo Methods to Train Neural Network Models," *Neural Computation*, vol. 12, no. 4, pp. 955–993, 2000.

[23] N. J. Gordon, D. J. Salmond, and A. F. M. Smith, "Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation," in *IEE Proceedings F (Radar and Signal Processing)*. IET, 1993, vol. 140, pp. 107–113.

[24] A. Doucet, N. De Freitas, and N. Gordon, "An Introduction to Sequential Monte Carlo Methods," in *Sequential Monte Carlo Methods in Practice*, pp. 3–14. Springer, 2001.

[25] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov Chain Monte Carlo Methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 3, pp. 269–342, 2010.

[26] X. R. Li and V. P. Jilkov, "Survey of Maneuvering Target Tracking. Part I. Dynamic Models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333–1364, 2003.

[27] D. Crisan and A. Doucet, "A Survey of Convergence Results on Particle Filtering Methods for Practitioners," *IEEE Transactions on signal processing*, vol. 50, no. 3, pp. 736–746, 2002.

[28] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[29] A. Doucet and A. M. Johansen, "A Tutorial on Particle Filtering and Smoothing: Fifteen Years Later," 2009.

[30] T. Li, M. Bolic, and P. M. Djuric, "Resampling Methods for Particle Filtering: Classification, Implementation, and Strategies," *IEEE Signal Processing Magazine*, vol. 32, no. 3, pp. 70–86, 2015.

[31] L. Deng, "The Mnist Database of Handwritten Digit Images for Machine Learning Research [best of the web]," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[32] A. Nguyen, J. Yosinski, and J. Clune, "Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2015, pp. 427–436.

[33] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and Harnessing Adversarial Examples," *arXiv preprint arXiv:1412.6572*, 2014.