ROBUST LEARNING VIA ENSEMBLE DENSITY PROPAGATION IN DEEP NEURAL NETWORKS

Giuseppina Carannante[⋆], *Dimah Dera*[⋆], *Ghulam Rasool*[⋆], *Nidhal C. Bouaynaya*[⋆], *and Lyudmila Mihaylova*[†]

[⋆] Rowan University, Department of Electrical and Computer Engineering, Glassboro, NJ
[†] University of Sheffield, Department of Automatic Control and Systems Engineering, United Kingdom
[⋆]carannang1@rowan.edu, [⋆]derad6@rowan.edu, [⋆]rasool@rowan.edu,
[⋆]bouaynaya@rowan.edu, [†]l.s.mihaylova@sheffield.ac.uk

ABSTRACT

Learning in uncertain, noisy, or adversarial environments is a challenging task for deep neural networks (DNNs). We propose a new theoretically grounded and efficient approach for robust learning that builds upon Bayesian estimation and Variational Inference. We formulate the problem of density propagation through layers of a DNN and solve it using an Ensemble Density Propagation (EnDP) scheme. The EnDP approach allows us to propagate moments of the variational probability distribution across the layers of a Bayesian DNN, enabling the estimation of the mean and covariance of the predictive distribution at the output of the model. Our experiments using MNIST and CIFAR-10 datasets show a significant improvement in the robustness of the trained models to random noise and adversarial attacks.

Index Terms— Variational inference, Ensemble techniques, robustness, adversarial learning.

1. INTRODUCTION

Recently, machine learning models have shown significant success in various application areas, including computer vision and natural language processing [1, 2]. However, these models may have limited suitability for mission-critical real-world applications due to the lack of information about the uncertainty (or equivalently confidence) in their predictions [3]. Information about uncertainty and confidence can improve a model's robustness to random noise and adversarial attacks [4, 5]. Many real-world applications, including various autonomous, military, or medical diagnosis and treatment systems, require the estimation of a model's confidence in its decisions [4, 5]. Quantitative estimation of uncertainty in the model's prediction can be accomplished by exploiting well-established Bayesian methods.

In Bayesian settings, we start by defining a prior probability distribution over the unknown parameters, i.e., weights and biases of a DNN. Bayes' theorem allows us to infer the posterior distribution of these parameters after observing the training data [6, 7, 8]. However, inferring the exact posterior distribution is mathematically intractable for most modern DNNs, as these models do not lend themselves to exact integration due to a large parameter space and multiple layers of nonlinearities [9]. One of the most common scalable density approximation approaches is Variational Inference (VI). The VI approximation method converts the intractable density inference into an optimization problem that is solved using standard algorithms, e.g., gradient descent [9, 7]. VI methods pose a simple family of distributions over the unknown parameters and then find (through optimization) a member of this family that is closest, in terms of Kullback-Leibler (KL) divergence, to the desired posterior distribution [10]. Over the past few years, VI has been used to estimate the posterior distribution for fully-connected neural networks, convolutional neural networks (CNNs), and recurrent neural networks [11, 12, 6].

However, current Bayesian approaches based on VI do not propagate the variational distribution from one layer of the DNN to the next layer [11]. Instead, a single set of parameters is sampled from the variational posterior and is used in the forward pass [11]. Alternatively, the dropout is used at test time, mimicking a Bernulli distribution for the weights, to generate various samples, which, in turn, are used to calculate uncertainty in the output using the frequentist approach [6].

Recently, Dera *et al.* proposed a scalable and efficient approach, called extended VI (eVI), to propagate the first and second moments of the variational distribution through all layers of a CNN [8, 13]. Their method provided a mean vector and a covariance matrix at the output, corresponding to the network's prediction and uncertainty, respectively [8]. The authors used first-order Taylor series approximation to compute the mean and covariance after propagating the variational distribution thorough the activation functions. However, the first-order Taylor series approximation may fail when the activation function is highly nonlinear, e.g., ELU, SELU, and Swish [14, 15, 16].

We build our Ensemble Density Propagation (EnDP)

978-1-7281-6662-9/20/$31.00 ©2020 IEEE

framework using the powerful statistical technique developed for density tracking in Ensemble Kalman Filters [17]. We propagate random samples across the layers of DNNs and estimate the first two moments of the variational posterior after passing through each layer, including nonlinear activation functions. Our results show that propagating the variational posterior using EnDP results in increased robustness to Gaussian noise and adversarial attacks.

The rest of this paper is structured in the following way. In section 2, we describe the general VI framework and introduce our proposed Ensemble Density Propagation (enDP) approach. EnDP results in the propagation of uncertainty information from the input, as well as networks parameters, to the network output. In section 3, we present our results on a classification task using the MNIST and CIFAR-10 datasets and compare them to state-of-the-art VI approaches. In section 4, we discuss our results and present the effect of the ensemble size (number of random samples N) on the performance of the proposed EnDP approach.

2. ENSEMBLE DENSITY PROPAGATION

A framework for the propagation of the variational posterior density across layers of DNNs has been recently explored [8]. In this paper, we introduce the Ensemble Density Propagation framework for tracking moments across layers of DNNs. We adopt the stochastic ensemble framework, drawing upon the ensemble Kalman filter and other Monte Carlo approaches [18, 19].

We define a prior probability distribution $p(\Omega)$ over the set of weights Ω of a DNN. After observing the training dataset $\mathcal{D}$, we update our belief and find the posterior distribution $p(\Omega|\mathcal{D})$. As the direct inference of $p(\Omega|\mathcal{D})$ is intractable, we employ VI to approximate the true posterior with a parametrized distribution $q_\theta(\Omega)$, also known as the variational posterior, with $\boldsymbol{\theta}$ representing the distribution parameters [10]. We assume $q_{\boldsymbol{\theta}}(\Omega)$ to be a Gaussian distribution. In VI, we minimize the KL-divergence between the true and the variational posterior distribution:

$$\mathbf{KL}(q_{\boldsymbol{\theta}}(\Omega)\big|\big|p(\Omega|\mathcal{D})) = \int q_{\boldsymbol{\theta}}(\Omega)\log\frac{q_{\boldsymbol{\theta}}(\Omega)}{p(\Omega)p(\mathcal{D}|\Omega)}d\Omega. \quad (1)$$

By rearranging the terms in (1), we obtain the following objective function:

$$\mathcal{L}(\boldsymbol{\theta}) = -\,\mathbb{E}_{q_{\boldsymbol{\theta}}(\boldsymbol{\Omega})}[\log(p(\mathrm{D}|\Omega)] + \mathbf{KL}(q_{\boldsymbol{\theta}}(\Omega)\big|\big|p(\Omega)), \quad (2)$$

where $\mathcal{L}(\boldsymbol{\theta})$ is widely known as the *variational free energy* and is composed of two terms, the expected log-likelihood, which depends on the data, and the KL-divergence between the prior and variational posterior, which does not depend on the data and acts as a regularization penalty. For simplicity and without loss of generality, we present our EnDP framework for a single layer CNN with one max-pooling layer and a fully connected layer before the soft-max function.

Convolution Operation: In our framework, the convolutional kernels are assumed to be random variables endowed with a multivariate Gaussian distribution. We assume that the kernels within a convolutional layer are independent of each other. This assumption reduces the number of unknown parameters and also forces convolutional kernels to extract features that are uncorrelated with each other.

We consider the convolution operation as a matrix-vector multiplication. We express the output of the convolutional layer as $\mathbf{z}^{(k_c)} = \mathbf{X}\ \mathrm{vec}(\boldsymbol{\mathcal{W}}^{(k_c)})$, $k_c = 1, \cdots, K_c$, where $\mathbf{X}$ represents a matrix whose rows are the vectorized sub-tensors of the input image, $\boldsymbol{\mathcal{W}}^{(k_c)}$ is the k_c^{th} convolutional kernel with $\mathrm{vec}(\boldsymbol{\mathcal{W}}^{(k_c)}) \sim \mathcal{N}\left(\mathbf{m}^{(k_c)}, \boldsymbol{\Sigma}^{(k_c)}\right)$, K_c is the total number of kernels and (vec) is the vectorization operation. Thus, the output of the convolution between the k_c^{th} kernel and the input image has a distribution $\mathbf{z}^{(k_c)} \sim \mathcal{N}\left(\mathbf{X}\mathbf{m}^{(k_c)},\ \mathbf{X}\boldsymbol{\Sigma}^{(k_c)}\mathbf{X}^T\right)$.

Nonlinear Activation Function: After the convolution, the resulting random variables $\mathbf{z}^{(k_c)}$ will be propagated through an element-wise nonlinear activation function ψ. We perform stochastic sampling and draw N samples, $\mathbf{z}_i^{(k_c)}$, where $i = 1, 2, \cdots, N$. We pass each ensemble member $\mathbf{z}_i^{(k_c)}$ through the activation function and obtain $\mathbf{g}_i^{(k_c)} = \psi[\mathbf{z}_i^{(k_c)}]$. We find the sample mean and covariance of $\mathbf{g}^{(k_c)}$ using:

$$\boldsymbol{\mu}_{\mathbf{g}^{(k_c)}} = \frac{1}{n}\sum_{i=1}^{N}\mathbf{g}_i^{(k_c)}, \quad (3)$$

$$\boldsymbol{\Sigma}_{\mathbf{g}^{(k_c)}} = \frac{1}{n-1}\sum_{i=1}^{N}\left[\mathbf{g}_i^{(k_c)} - \boldsymbol{\mu}_{\mathbf{g}^{(k_c)}}\right]\left[\mathbf{g}_i^{(k_c)} - \boldsymbol{\mu}_{\mathbf{g}^{(k_c)}}\right]^T.$$

Max-Pooling Operation: The max-pooling operation selects the largest value in each patch of the given input. At the output of the max-pooling layer, we approximate the mean by $\boldsymbol{\mu}_{\mathbf{p}^{(k_c)}} = \mathrm{pool}(\boldsymbol{\mu}_{\mathbf{g}^{(k_c)}})$. For the covariance matrix, we keep rows and columns of $\boldsymbol{\Sigma}_{\mathbf{g}^{(k_c)}}$ corresponding to the elements of the mean vector retained after the pooling operation, i.e., $\boldsymbol{\Sigma}_{\mathbf{p}^{(k_c)}} = \mathrm{pool}(\boldsymbol{\Sigma}_{\mathbf{g}^{(k_c)}})$. If we denote by $d_1 \times d_1$ the dimension of $\mathbf{g}^{(k_c)}$. Thus, $\boldsymbol{\mu}_{\mathbf{g}^{(k_c)}}$ has the same dimension as $\mathbf{g}^{(k_c)}$ and $\boldsymbol{\Sigma}_{\mathbf{g}^{(k_c)}}$ has a dimension $d_1^2 \times d_1^2$. At the output of max-pooling, the dimensions of $\boldsymbol{\mu}_{\mathbf{p}^{(k_c)}}$, and $\boldsymbol{\Sigma}_{\mathbf{p}^{(k_c)}}$ become $d_2 \times d_2$ and $d_2^2 \times d_2^2$, respectively, where $d_2 = (d_1 - p)/s + 1$, p is the patch size of the pooling operation and s is the stride.

Fully-Connected (FC) Layer: The input to the FC layer $\mathbf{b}$ is obtained by vectorizing the output of the max-pooling layer. The mean and covariance of $\mathbf{b}$ are given by:

$$\boldsymbol{\mu}_{\mathbf{b}} = \begin{bmatrix}\boldsymbol{\mu}_{\mathbf{p}^{(1)}}\\ \vdots \\ \boldsymbol{\mu}_{\mathbf{p}^{(K_c)}}\end{bmatrix}, \boldsymbol{\Sigma}_{\mathbf{b}} = \begin{bmatrix}\boldsymbol{\Sigma}_{\mathbf{p}^{(1)}} & \cdots & \mathbf{0}\\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \boldsymbol{\Sigma}_{\mathbf{p}^{(K_c)}}\end{bmatrix} \quad (4)$$

We denote the hth weight vector of the FC layer by $\mathbf{w}_h \sim \mathcal{N}(\mathbf{m}_h, \boldsymbol{\Sigma}_h)$, for $h = 1, \cdots, H$, where H is the number of output neurons. By employing the derivations in [8] for the product of random vectors, we can compute the output mean, $\boldsymbol{\mu}_{\mathbf{f}}$, and the output covariance, $\boldsymbol{\Sigma}_{\mathbf{f}}$, of the FC-layer as:

$$\mu_{f_h} = \mathbf{m}_h^T \boldsymbol{\mu}_{\mathbf{b}}, \tag{5}$$

$$\boldsymbol{\Sigma}_{\mathbf{f}} = \begin{cases} \operatorname{Tr}\left(\boldsymbol{\Sigma}_{h_i}\boldsymbol{\Sigma}_{\mathbf{b}}\right) + \mathbf{m}_{h_i}^T \boldsymbol{\Sigma}_{\mathbf{b}} \mathbf{m}_{h_j} + \boldsymbol{\mu}_{\mathbf{b}}^T \boldsymbol{\Sigma}_{h_j} \boldsymbol{\mu}_{\mathbf{b}}, & i = j. \\ \mathbf{m}_{h_i}^T \boldsymbol{\Sigma}_{\mathbf{b}} \mathbf{m}_{h_j}, & i \neq j. \end{cases}$$

where $h, h_i, h_j = 1, 2, \cdots, H$, and i, j refer to any two weight vectors in the FC layer.

Soft-max Function: For multi-class problems, the network output is given by the soft-max function, i.e., $\hat{\boldsymbol{y}} = \phi(\boldsymbol{f})$, where ϕ represents the softmax function and $\boldsymbol{f}$ is the output of the FC layer. We can approximate the mean $\boldsymbol{\mu}_{\mathbf{y}}$ and covariance $\boldsymbol{\Sigma}_{\mathbf{y}}$ using first-order Taylor series approximation:

$$\boldsymbol{\mu}_y \approx \phi(\boldsymbol{\mu}_f), \text{ and } \boldsymbol{\Sigma}_{\mathbf{y}} \approx \boldsymbol{J}_\phi \boldsymbol{\Sigma}_{\mathbf{f}} \boldsymbol{J}_\phi^T, \tag{6}$$

where $\boldsymbol{J}_\phi$ represents the Jacobian matrix of ϕ with respect to $\boldsymbol{f}$ evaluated at $\boldsymbol{\mu}_f$. The proposed EnDP framework can be easily extended to multi-layer CNNs and various architectures (such as recurrent neural networks) by following the same derivation presented above.

3. EXPERIMENTS AND RESULTS

We evaluated the performance of the proposed EnDP method on a classification task, using two datasets, i.e., MNIST handwritten digits and CIFAR-10 [20, 21]. We compared test accuracy of our model with the state-of-the-art in the literature, including a vanilla CNN, Bayes-by-Backprop (BBB), Bayes-CNN, Dropout-CNN, and eVI [11, 12, 6, 8]. We evaluated all models using test datasets of MNIST and CIFAR-10 under three conditions, i.e., noise-free, Gaussian noise, and adversarial attack. The targeted adversarial examples were generated using the fast gradient sign method (FGSM) [22].

3.1. MNIST Dataset

We used a CNN having one convolutional layer with 32 kernels of size 5×5, followed by the rectified linear unit (ReLU) activation, one max-pooling layer and one FC layer. We used $N = 1000$ samples for the ensemble density propagation. We tested all models at two levels of Gaussian noise, i.e., $\sigma^2_{\text{noise}} = 0.1$, and 0.2. The adversarial examples were generated to fool each model into predicting digit "3" with two attack levels, i.e., $\sigma_{\text{adversarial}} = 0.1$, and 0.2.

In Table 1, we present test accuracies of EnDP, eVI, BBB, and a vanilla CNN for the MNIST test set at various levels of Gaussian noise and adversarial attacks. In Fig. 1, we present selected test results of EnDP for three different noise conditions, i.e., noise-free, Gaussian noise, and adversarial attack. We present test images with their ground-truth and predicted labels, and corresponding outputs of the soft-max function (the mean vectors $\boldsymbol{\mu}_y$ and covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{y}}$ from Eq. 6). The diagonal elements of the covariance matrix, i.e., the variance elements, provide a meaningful and calibrated measure of the model's uncertainty or equivalently confidence associated with every prediction.

In Fig. 2, we present the test accuracy and training time of EnDP for various sample sizes N used for ensemble density propagation.

3.2. CIFAR-10 Dataset

We used a CNN with three convolutional blocks and one FC layer. Each convolutional block included two consecutive convolutional layers, each followed by Exponential Linear Unit (ELU) activation function and one max-pooling layer at the end [15]. The convolutional kernels in all blocks were of size 3×3. The number of convolutional kernels in the first, second and third block was set to 32, 64, and 128, respectively. In total, our network included six convolutional layers, each followed by ELU activation.

For the ensemble density propagation, we used a different number of samples for each of the six ELU layers, i.e., $N_i = 2d_i$, where $i = 1, 2, \ldots, 6$ represent ELU layers, and d_i is the dimension of the feature map obtained after the ith convolutional layer. In Table 2, we report test accuracy of EnDP, eVI, Bayes-CNN and Dropout-CNN for the noise-free case and under adversarial and Gaussian noise conditions. The noise level was set to 5% of the the highest conceivable value (HCV), where HCV $= 3\ \sigma_{\text{noise}}$ [23]. We generated the targeted adversarial examples to fool each network into predicting the label "cat".

4. DISCUSSION

We proposed a new method for propagating variational posterior distribution through nonlinear activation functions in DNNs using the ensemble approach. We draw N random samples, pass these samples thought the nonlinear activation functions, and calculate the mean and covariance of the transformed output. The propagation of the distribution through

Table 1. MNIST Test Accuracy

Noise/Attack level	EnDP	eVI	BBB	CNN
No Noise	97%	96%	96%	96%
Gaussian Noise				
0.1	95%	94%	86%	79%
0.2	86%	85%	76%	70%
Adversarial Attack				
0.1	95%	95%	91%	58%
0.2	83%	81%	45%	14%

(a) Noise Free

True: 2, Pred: 2

$\boldsymbol{\mu}_y$

	Mean
0	0.00157
1	1.15E-06
2	0.99840
3	5.96E-06
4	1.06E-08
5	3.20E-07
6	2.97E-08
7	0.00002
8	5.03E-08
9	2.11E-08

$\boldsymbol{\Sigma}_y$

	0	1	2	3	4	5	6	7	8	9
0	0.00125	4.64E-07	-0.00126	2.39E-06	4.27E-09	1.29E-07	1.19E-08	8.80E-06	2.02E-08	8.50E-09
1	4.64E-07	6.76E-10	-4.73E-07	1.78E-09	3.16E-12	9.54E-11	8.74E-12	6.50E-09	1.48E-11	6.27E-12
2	-0.00126	-4.73E-07	0.00127	-2.44E-06	-4.35E-09	-1.31E-07	-1.21E-08	-9.09E-06	-2.05E-08	-8.66E-09
3	2.39E-06	1.78E-09	-2.44E-06	1.84E-08	1.63E-11	4.98E-10	4.53E-11	3.37E-08	7.64E-11	3.27E-11
4	4.27E-09	3.16E-12	-4.35E-09	1.63E-11	5.79E-14	8.77E-13	8.08E-14	6.00E-11	1.37E-13	5.76E-14
5	1.29E-07	9.54E-11	-1.31E-07	4.98E-10	8.77E-13	5.27E-11	2.44E-12	1.81E-09	4.10E-12	1.75E-12
6	1.19E-08	8.74E-12	-1.21E-08	4.53E-11	8.08E-14	2.44E-12	4.47E-13	1.66E-10	3.81E-13	1.60E-13
7	8.80E-06	6.50E-09	-9.09E-06	3.37E-08	6.00E-11	1.81E-09	1.66E-10	2.48E-07	2.81E-10	1.19E-10
8	2.02E-08	1.48E-11	-2.05E-08	7.64E-11	1.37E-13	4.10E-12	3.81E-13	2.81E-10	1.29E-12	2.73E-13
9	8.50E-09	6.27E-12	-8.66E-09	3.27E-11	5.76E-14	1.75E-12	1.60E-13	1.19E-10	2.73E-13	2.28E-13

(b) Gaussian Noise

True: 2, Pred: 2

	Mean
0	0.00548
1	0.00020
2	0.98329
3	0.00628
4	4.87E-05
5	2.73E-05
6	1.58E-05
7	0.00328
8	0.00057
9	0.00081

	0	1	2	3	4	5	6	7	8	9
0	0.00767	0.00014	-0.01544	0.00433	3.39E-05	1.91E-05	1.10E-05	0.00227	0.00040	0.00056
1	0.00014	9.97E-06	-0.00042	0.00016	1.23E-06	6.94E-07	3.98E-07	8.22E-05	1.43E-05	2.04E-05
2	-0.01544	-0.00042	0.04603	-0.01855	-0.00010	-5.88E-05	-3.37E-05	-0.00836	-0.00125	-0.00181
3	0.00433	0.00016	-0.01855	0.01027	3.91E-05	2.22E-05	1.27E-05	0.00261	0.00045	0.00065
4	3.39E-05	1.23E-06	-0.00010	3.91E-05	6.15E-07	1.72E-07	9.88E-08	2.04E-05	3.56E-06	5.05E-06
5	1.91E-05	6.94E-07	-5.88E-05	2.22E-05	1.72E-07	1.96E-07	5.59E-08	1.15E-05	1.99E-06	2.85E-06
6	1.10E-05	3.98E-07	-3.37E-05	1.27E-05	9.88E-08	5.59E-08	6.49E-08	6.60E-06	1.16E-06	1.64E-06
7	0.00227	8.22E-05	-0.00836	0.00261	2.04E-05	1.15E-05	6.60E-06	0.00278	0.00024	0.00034
8	0.00040	1.43E-05	-0.00125	0.00045	3.56E-06	1.99E-06	1.16E-06	0.00024	8.46E-05	5.91E-05
9	0.00056	2.04E-05	-0.00181	0.00065	5.05E-06	2.85E-06	1.64E-06	0.00034	5.91E-05	0.00017

(c) Adversarial Noise

True: 2, Pred: 3

	Mean
0	0.00101
1	0.00011
2	0.15752
3	0.81976
4	2.03E-05
5	2.82E-05
6	5.25E-06
7	0.01940
8	0.00076
9	0.00140

	0	1	2	3	4	5	6	7	8	9
0	0.00030	1.33E-05	0.01507	-0.01794	2.49E-06	3.39E-06	6.47E-07	0.00229	9.52E-05	0.00017
1	1.33E-05	3.46E-06	0.00161	-0.00190	2.70E-07	3.69E-07	6.96E-08	2.48E-04	1.02E-05	1.83E-05
2	0.01507	0.00161	6.04635	-6.37309	0.00030	0.00041	7.85E-05	0.27741	0.01147	0.02039
3	-0.01794	-0.00190	-6.37309	6.82484	-0.00036	-0.00048	-9.26E-05	-0.39292	-0.01362	-0.02443
4	2.49E-06	2.70E-07	0.00030	-0.00036	1.22E-07	6.87E-08	1.31E-08	4.65E-05	1.90E-06	3.42E-06
5	3.39E-06	3.69E-07	0.00041	-0.00048	6.87E-08	2.33E-07	1.80E-08	6.37E-05	2.56E-06	4.70E-06
6	6.47E-07	6.96E-08	7.85E-05	-9.26E-05	1.31E-08	1.80E-08	8.10E-09	1.20E-05	4.91E-07	8.86E-07
7	0.00229	0.00025	0.27741	-0.39292	4.65E-05	6.37E-05	1.20E-05	0.10796	0.00174	0.00314
8	9.52E-05	1.02E-05	0.01147	-0.01362	1.90E-06	2.56E-06	4.91E-07	0.00174	0.00017	0.00013
9	0.00017	1.83E-05	0.02039	-0.02443	3.42E-06	4.70E-06	8.86E-07	0.00314	0.00013	0.00057

Fig. 1. The output of the EnDP model, i.e., the mean vector $\boldsymbol{\mu}_y$ and covariance matrix $\boldsymbol{\Sigma}_y$ of the soft-max function, is presented for three test images. In sub-figures (b) and (c), test images were corrupted with Gaussian noise ($\sigma^2_{\text{noise}} = 0.1$) and adversarial attack ($\sigma_{\text{adversarial}} = 0.1$), respectively. The green color refers to the predicted output, while the yellow color represents the ground truth. When the yellow block is not shown, the network prediction and the ground-truth labels matched. In the covariance matrix, a large variance value indicates a low level of confidence or high uncertainty in the prediction.

DNNs results in robust performance against Gaussian noise and adversarial attacks.

In the noise-free case, our approach, referred to as EnDP, performed better or equally on two benchmark datasets (MNIST and CIFAR-10) as compared to the state-of-the-art models, including eVI, BBB, Bayes-CNN, Dropout-CNN, and a vanilla CNN. Under noisy conditions and adversarial attacks, EnDP outperformed all models (except for the MNIST dataset at a low level of adversarial attack where EnDP and eVI produced 95% test accuracy, Table 1). We note that as the level of noise or severity of adversarial attack increased (Table 1), the EnDP model maintained better performance. The gap between the accuracy of EnDP and other models increased. Similarly, in relatively complex network architecture (CIFAR-10 dataset, Table 2), EnDP performed robustly as compared to all other models in noise-free conditions as well as under noise attack.

4.1. Effect of Sample Size (N)

We note that both the accuracy and training time increase with the increasing number of samples used for ensemble density propagation (Fig. 2). This behavior agrees with the well-known trade-off between accuracy and computational cost. Our empirical results show that the number of samples required to achieve comparable accuracy depends upon the size of the feature map resulting from the preceding convolutional layer. We found that the number of samples approximately equal to twice the size of the feature map produced good results. For the case of MNIST, the output of the convolution operation $\mathbf{z}$ is of size $d = 24 \times 24 = 576$. Therefore, we used $N = 1000$ for our experiments, which resulted in comparable accuracy at a reasonable computational cost. For CIFAR-10, we vary N for each ELU layer depending upon the size of the output of the preceding convolutional layer ($N_i = 2d_i$).

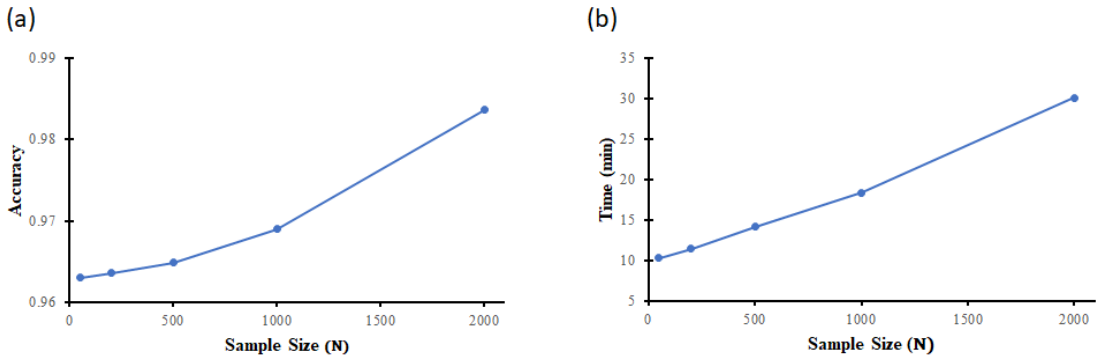

Fig. 2. The effect of number of samples N used for EnDP on the test accuracy and training time for MNIST dataset is presented. (a) Test accuracy increases as N increases. (b) Training time (in minutes) for one epoch as N is increased.

4.2. Robustness to Noise and Adversarial Attacks

We consider that the robustness of EnDP models to noise and adversarial attacks is attributable to the propagation of moments of the variational posterior through the network layers. The propagation of moments enables the model to use confidence (i.e., variance/covariance) information during the optimization process. In the moment propagation settings, the network learns "robust" parameters, including convolutional kernels and weights of the FC layer. The learned "robust" parameters result in a robust behavior, especially when the input is corrupted with noise or is adversarially attacked.

Both EnDP and eVI are based on variational posterior density propagation and show robustness in noisy and adversarial environments. However, the proposed EnDP method is superior to eVI, as evident in the experimental results, especially at a high level of noise and adversarial attacks. In our experiments, we used two activation functions, ReLU and ELU. However, the EnDP framework is readily extendable to all types of activation functions. Owing to the sampling and stochastic nature of our proposed EnDP technique, we consider that the performance of EnDP will be even better for highly nonlinear activation functions. In fact, we expect that for highly nonlinear activation functions (e.g., Gaussian Error Linear Unit, and Scaled exponential linear unit), the first-order approximation used in eVI might fail since higher-order terms are neglected in the linearisation; however, the proposed EnDP technique will perform robustly.

Table 2. CIFAR-10 Test Accuracy

Noise Type	EnDP	eVI	Bayes-CNN	Dropout-CNN
Zero	86%	86%	85%	86%
Adversarial	82%	80%	68%	52%
Gaussian	85%	82%	77%	75%

4.3. Calibrated Uncertainty Information in the Model's Predictions

The predictions of modern neural networks (i.e., the output of the soft-max function) are poorly calibrated and may provide misleading interpretation, especially when the predicted output is wrong [24, 25]. A key feature of the proposed EnDP method is the availability of uncertainty information at the output through the covariance matrix. For example, consider the adversarial attack case in Fig. 1(c). The EnDP model erroneously predicted digit "3" instead of "2"; however, the variance values (diagonal elements) corresponding to digits "3" and "2" were significantly larger as compared to all others. If we set the confidence proportional to the inverse of the variance, the mentioned example revels that the EnDP model was highly uncertain about its prediction and indicating low confidence in its output. The availability of a calibrated measure, i.e., the covariance matrix, can help establish the trustworthiness of machine learning models. Furthermore, the variance information can provide insights that can help interpret a model's correct and incorrect predictions.

5. CONCLUSION

We proposed a new approach for the approximation of variational posterior in DNNs. We were able to propagate the first two moments of the variational posterior through the layers of a multi-layer CNN using a stochastic ensemble technique. The proposed Ensemble Density Propagation (EnDP) framework can approximate any number of moments. The covariance matrix available at the output of an EnDP model captures its uncertainty in the predicted decisions. Our experimental results using the MNIST and CIFAR-10 datasets showed significantly increased robustness of the EnDP models to Gaussian noise and adversarial attacks. We consider that the propagation of moments through layers of the network results in robust learning and improved performance in noisy conditions.

6. ACKNOWLEDGEMENT

This work was supported by the National Science Foundation Awards NSF ECCS-1903466, NSF CCF-1527822, NSF OAC-2008690. Giuseppina Carannante is supported by the US Department of Education through a Graduate Assistance in Areas of National Need (GAANN) Program Award Number P200A180055. We are also grateful to UK EPSRC support through EP/T013265/1 project NSF-EPSRC: ShiRAS. Towards Safe and Reliable Autonomy in Sensor Driven Systems.

7. REFERENCES

[1] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 1097–1105.

[2] Tero Karras, Samuli Laine, and Timo Aila, "A Style-Based Generator Architecture for Generative Adversarial Networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[3] Alex Kendall and Yarin Gal, "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, 2017, pp. 5574–5584.

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

[5] Justin Ker, Lipo Wang, Jai Rao, and Tchoyoson Lim, "Deep Learning Applications in Medical Image Analysis," *Ieee Access*, vol. 6, pp. 9375–9389, 2017.

[6] Yarin Gal and Zoubin Ghahramani, "Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference," in *Proceedings of 4th International Conference on Learning Representations, workshop track*, 2016.

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

[8] Dimah Dera, Ghulam Rasool, and Nidhal Bouaynaya, "Extended Variational Inference for Propagating Uncertainty in Convolutional Neural Networks," in *Proceedings of the IEEE International Workshop on Machine Learning for Signal Processing*, 2019.

[9] Christopher M Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[10] Geoffrey Hinton and Drew Van Camp, "Keeping Neural Networks Simple by Minimizing the Description Length of the Weights," in *in Proc. of the 6th Ann. ACM Conf. on Computational Learning Theory*. Citeseer, 1993.

[11] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra, "Weight Uncertainty in Neural Networks," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, 2015, vol. 37, pp. 1613–1622.

[12] Kumar Shridhar, Felix Laumann, Adrian Llopart Maurin, and Marcus Liwicki, "Bayesian Convolutional Neural Networks," *arXiv preprint arXiv:1806.05978*, 2018.

[13] Dimah Dera, Ghulam Rasool, Nidhal C. Bouaynaya, Adam Eichen, Stephen Shanko, Jeff Cammerata, and Sanipa Arnold, "Bayes-SAR Net: Robust SAR Image Classification with Uncertainty Estimation using Bayesian Convolutional Neural Network," in *Proceedings of the IEEE International Radar Conference*, 2020.

[14] Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter, "Self-Normalizing Neural Networks," in *Advances in Neural Information Processing Systems*, 2017, pp. 971–980.

[15] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter, "Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)," in *Proceedings of 4th International Conference on Learning Representations*, 2016.

[16] Prajit Ramachandran, Barret Zoph, and Quoc V Le, "Swish: a Self-Gated Activation Function," *arXiv preprint arXiv:1710.05941*, vol. 7, 2017.

[17] Simon J Julier and Jeffrey K Uhlmann, "New Extension of the Kalman Filter to Nonlinear Systems," in *Signal processing, sensor fusion, and target recognition VI*. International Society for Optics and Photonics, 1997, vol. 3068, pp. 182–193.

[18] Huazhen Fang, Ning Tian, Yebin Wang, MengChu Zhou, and Mulugeta A Haile, "Nonlinear Bayesian Estimation: From Kalman Filtering to a Broader Horizon," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 2, pp. 401–417, 2018.

[19] Rudolph Van Der Merwe, Arnaud Doucet, Nando De Freitas, and Eric A Wan, "The Unscented Particle Filter," in *Advances in Neural Information Processing Systems*, 2001, pp. 584–590.

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

[21] A. Krizhevsky, "Learning Multiple Layers of Features from Tiny Images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009.

[22] Y. Liu, X. Chen, C. Liu, and D. Song, "Delving into Transferable Adversarial Examples and Black-Box Attacks," in *Proceedings of 5th International Conference on Learning Representations*, 2017.

[23] J Michael Duncan, "Factors of Safety and Reliability in Geotechnical Engineering," *Journal of Geotechnical and Geoenvironmental Engineering*, vol. 126, no. 4, pp. 307–316, 2000.

[24] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger, "On Calibration of Modern Neural Networks," in *Proceedings of the 34th International Conference on Machine Learning-Volume*, 2017, pp. 1321–1330.

[25] Anh Nguyen, Jason Yosinski, and Jeff 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.