PremiUm-CNN: Propagating Uncertainty Towards Robust Convolutional Neural Networks

Dimah Dera, *Member, IEEE,* Nidhal C. Bouaynaya, *Member, IEEE,* Ghulam Rasool, *Member, IEEE,* Roman Shterenberg, and Hassan M. Fathallah-Shaykh,

***Abstract*—Deep neural networks (DNNs) have surpassed human-level accuracy in various learning tasks. However, unlike humans who have a natural cognitive intuition for probabilities, DNNs cannot express their uncertainty in the output decisions. This limits the deployment of DNNs in mission-critical domains, such as warfighter decision-making or medical diagnosis. Bayesian inference provides a principled approach to reason about model's uncertainty by estimating the posterior distribution of the unknown parameters. The challenge in DNNs remains the multi-layer stages of non-linearities, which make the propagation of high-dimensional distributions mathematically intractable. This paper establishes the theoretical and algorithmic foundations of uncertainty or belief propagation by developing new deep learning models named PremiUm-CNNs (Propagating Uncertainty in Convolutional Neural Networks). We introduce a tensor normal distribution as a prior over convolutional kernels and estimate the variational posterior by maximizing the evidence lower bound (ELBO). We start by deriving the first-order mean-covariance propagation framework. Later, we develop a framework based on the unscented transformation (correct at least up to the second-order) that propagates *sigma points* of the variational distribution through layers of a CNN. The propagated covariance of the predictive distribution captures uncertainty in the output decision. Comprehensive experiments conducted on diverse benchmark datasets demonstrate: 1) superior robustness against noise and adversarial attacks, 2) self-assessment through predictive uncertainty that increases quickly with increasing levels of noise or attacks, and 3) an ability to detect a targeted attack from ambient noise.**

***Index Terms*—Density propagation, first-order approximation, *sigma points*, convolutional neural network (CNN), evidence lower bound (ELBO), tensor normal distribution (TND).**

I. Introduction

DEEP neural networks (DNNs) have achieved state-of-the-art performance in a wide assortment of tasks, including computer vision and pattern recognition [1], [2]. However, DNNs being inherently deterministic, are unable to provide calibrated confidence or a measure of uncertainty in their predictions [3]–[7]. The soft-max output is often misinterpreted as a measure of confidence because the soft-max function transforms its domain values from $(-\infty, \infty)$ to a range of $(0, 1)$. DNNs are known to assign high soft-max values to the wrong class when the input sample is far from the training distribution [8], [9].

Model confidence or uncertainty is critical in systems that make decisions that affect human life, either directly or indirectly. For example, a medical diagnostic system for detecting brain tumors from magnetic resonance (MR) scans may encounter a new tumor shape/structure or an adversarial attack designed to fake a tumor for benefiting from medical bills. The model should recognize the out-of-distribution data and return an output that also conveys a high level of uncertainty or low confidence in the decision. Failure to indicate when the networks are likely mistaken can lead to perilous consequences and limit their competence in applications such as automated medical diagnostic systems [10].

Besides safety-critical applications, the fundamental research questions in machine learning can also benefit from uncertain information, including (1) choosing the appropriate data to learn from and (2) searching for the optimal model architecture. Central to both questions is the uncertainty information available at the neural network's output, which can guide the learning process and help learn from a small amount of data. Quantifying uncertainty is often essential when obtaining labeled data is expensive due to the human expertise required to annotate data.

Bayesian probability theory provides a mathematically grounded approach for quantifying uncertainty of a model, including DNNs. In the Bayesian framework, model parameters, i.e., the weights and biases, are defined as random variables with a prior distribution. The posterior distribution of the parameters given available data is used to find the predictive distribution of new data instances by marginalizing the parameters. The covariance matrix of the predictive distribution provides a quantitative measure of uncertainty associated with each prediction. Therefore, estimating uncertainty in DNNs is directly dependent on estimating the posterior of unknown parameters given the data. However, the exact Bayesian inference on the parameters of a DNN is intractable as the functional form of a DNN does not lend itself to exact integration, and the number of parameters is very large [11]. Various approaches have been proposed to approximate the posterior distribution of the network's parameters, including the well-known variational inference (VI) [11]–[18]. The challenge remains in propagating the variational distribution through the non-linear layers of DNNs. VI methods in the literature avoid propagating density by following a sampling paradigm where one sample from the variational distribution is drawn randomly

Manuscript received February 16, 2021 and revised May 19, 2021.

D. Dera, G. Rasool and N. C. Bouaynaya are with the Department of Electrical and Computer Engineering, Rowan University, Glassboro, NJ, 08028 USA (e-mails: derad6@rowan.edu, rasool@rowan.edu and bouaynaya@rowan.edu).

R. Shterenberg is with the Department of Mathematics, University of Alabama at Birmingham, Birmingham, AL 35294-1170 USA (e-mail: shterenb@uab.edu).

H. M. Fathallah-Shaykh is with the Department of of Neurology, University of Alabama at Birmingham School of Medicine, Birmingham, AL 35294-1170 USA (e-mail: hfshaykh@uabmc.edu).

and passed forward through network layers [11], [17], [19].

In this paper, we propose a general framework, named PremiUm-CNN, that propagates the variational probability distribution through layers of a convolutional neural network (CNN) and enables the estimation of uncertainty at the output. PremiUm-CNN adopts powerful statistical frameworks from sequential Bayesian estimation for tracking distributions in non-linear and non-Gaussian dynamical systems [20]. The specific contributions of this paper are summarized as follows,

- We employ the first-order Taylor series approximation (termed Extended Variational Density Propagation, i.e., exVDP) for estimating the first two moments of the variational distribution after non-linear activation functions in DNNs (extension of our preliminary work in [21]).
- We develop the Unscented Variational Density Propagation, i.e., unVDP model for approximating the posterior distribution using the unscented transformation (UT). The UT propagates *sigma points* through the network's layers and results in a posterior distribution approximation that can tackle non-Gaussian distributions and is accurate at least up to the second order [20], [22], [23].
- We establish superior robustness by analyzing the models' performance (compared to the state-of-the-art DNNs' performance) under noisy conditions and adversarial attacks using a variety of benchmark datasets, including MNIST, CIFAR-10, Synthetic Aperture Radar (SAR), and Brain Tumor Segmentation (BraTS 2015) datasets.

The remainder of this paper is organized as follows. We briefly recall the state-of-the-art Bayesian approximation approaches for DNNs in Section II. In Section III, we describe the detail of the proposed PremiUm-CNN. We evaluate the performance and study the robustness and self-awareness of the proposed framework under noise and adversarial attacks in Sections IV and V, respectively. A summary of the main contributions is provided in Section VI.

II. Background and Related Work

Earlier efforts for integrating Bayesian inference into neural networks included Laplace approximation, which assumes that the posterior is a Gaussian distribution [24]. The mean of the posterior is given by the maximum a posteriori estimate and the covariance by the inverse of the Hessian of the negative log-likelihood, which is intractable for DNNs. Ritter *et al.* recently proposed a scalable approximation for estimating the Hessian; however, the Laplace approximation was employed at the test time only, i.e., the training was performed in a deterministic setting without learning uncertainty from the training dataset [16]. Another earlier effort included Hamiltonian Monte Carlo (HMC), i.e., a method based on Markov chain Monte Carlo (MCMC), for generating samples from the posterior distribution, which suffered from computational challenges [25]. Stochastic gradient MCMC was proposed to scale sampling methods to large datasets and DNNs using sub-samples of the dataset; nevertheless, the approximation efficiency remains questionable [26]–[28].

Expectation Propagation (EP) and assumed density filtering (ADF) are posterior approximation methods that iterate simple local computations to approximate factors of the posterior distribution for each data point [13], [14], [29]. Hernandez-Lobato and Adams proposed the probabilistic back-propagation (PBP), which applied approximate inference in the form of ADF to refine a Gaussian posterior approximation for regression problems [13]. Later, Ghosh *et al.* extended PBP to the multi-class classification problems [14]. The ADF approximation proposed by Hernandez-Lobato and Adams eliminated the dependence on ordering by doing multiple ADF passes over the data; however, the full EP implementation was impractical for DNNs due to massive computational and memory requirements [29].

Variational inference (VI) is a classical posterior density approximation technique that has been efficiently scaled to DNNs in recent years [11], [12], [15], [17]–[19], [30], [31]. Blundell *et al.* used VI and introduced a fully factorized Gaussian over the weights of the fully-connected layers of a DNN, referred to as the *Bayes-by-Backprop* (BBB) [11]. In a followup work, Shridhar *et al.* extended BBB to Bayes-CNN by proposing a fully factorized Gaussian distribution over convolutional kernels [17]. On the other hand, Gal and Ghahramani proposed Dropout CNNs using VI by interpreting dropout as a Bernoulli distribution over convolutional kernels [19]. Roth and Pernkopf proposed a closed-form approximation to the log-likelihood (first term in (2)) for fully-connected networks and ReLU activation [31]. Later, Wu *et al.* extended Roth and Pernkopf's work for the Heaviside activation function and developed an empirical Bayes method for tuning prior distribution during training [18]. Authors in [31] and [18] provided a closed-form approximation for the Gaussian posterior in fully-connected networks; however, the approximation was only limited to ReLU and Heaviside activation functions and did not perform well on classification problems as compared to the state-of-the-art. In [32], [33], the authors estimated observation (input signal) uncertainty using a deterministic pre-trained fully-connected neural network; however, they ignored the uncertainty in the network's parameters and its link to the output prediction.

In the VI-based methods, the moments of the variational distribution defined over the network parameters are generally *not propagated* from one layer of the DNN to the next layer. Only one sample is drawn randomly from the variational posterior and is passed forward through network layers [11], [17], [19]. The uncertainty in the output of the model is estimated using the frequentist approach, i.e., averaging stochastic forward passes through the model at test time using Monte Carlo and computing the sample variance [11], [17], [19].

III. PremiUm-CNN — Propagating Uncertainty in a Convolutional Neural Network

We consider a CNN with a total of C convolutional layers and L fully-connected layers. A non-linear activation function follows every convolutional and fully-connected layer. Moreover, every convolutional layer is followed by a max-pooling layer. The network's weights (and biases) are represented by $\mathbf{\Omega} = \{\{\{\mathcal{W}^{(k_c)}\}_{k_c=1}^{K_c}\}_{c=1}^{C}, \{\mathbf{W}^{(l)}\}_{l=1}^{L}\}$, where $\{\{\mathcal{W}^{(k_c)}\}_{k_c=1}^{K_c}\}_{c=1}^{C}$ is the set of K_c kernels in the c^{th} convolutional layer, and $\{\mathbf{W}^{(l)}\}_{l=1}^{L}$ is the set of weights in L fully-connected layers. We consider input tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times K}$,

where I_1, I_2, and K represent image height, width, and number of channels respectively.

A. Variational Inference

We introduce a prior distribution over the network weights $\mathbf{\Omega} \sim p(\mathbf{\Omega})$. We assume that convolutional kernels are independent of each other within a layer as well as across different layers; however, within a kernel, we assume that a covariance structure exits. This independence assumption is reasonable, perhaps even desirable as it promotes convolutional kernels to extract uncorrelated features within and across layers [34], [35]. Given the training data $\mathcal{D} = \{\mathcal{X}^{(i)}, \mathbf{y}^{(i)}\}_{i=1}^{N}$ and the prior $p(\mathbf{\Omega})$, the estimation of the posterior $p(\mathbf{\Omega}|\mathcal{D})$ is typically intractable. VI methods approximate the true posterior $p(\mathbf{\Omega}|\mathcal{D})$ with a simpler parametrized variational distribution $q_\phi(\mathbf{\Omega})$. The optimal parameters of the variational posterior ϕ^* are estimated by minimizing the Kullback-Leibler (KL) divergence between the approximate and the true posterior [36].

$$\begin{aligned}\phi^* &= \operatorname{argmin} \mathrm{KL}\left[q_\phi(\mathbf{\Omega}) \| p(\mathbf{\Omega}|\mathcal{D})\right] \\ &= \operatorname{argmin} \mathrm{KL}\left[q_\phi(\mathbf{\Omega}) \| p(\mathbf{\Omega})\right] - E_{q_\phi(\mathbf{\Omega})}\left\{\log p(\mathcal{D}|\mathbf{\Omega})\right\}.\end{aligned} \tag{1}$$

The optimization objective is given by the evidence lower bound (ELBO), $\mathfrak{L}(\phi; \mathbf{y}|\mathcal{X})$:

$$\mathfrak{L}(\phi; \mathbf{y}|\mathcal{X}) = E_{q_\phi(\mathbf{\Omega})}(\log p(\mathbf{y}|\mathcal{X}, \mathbf{\Omega})) - \mathrm{KL}\left[q_\phi(\mathbf{\Omega}) \| p(\mathbf{\Omega})\right]. \tag{2}$$

ELBO consists of two parts, the expected log-likelihood of the training data given the weights and a regularization term, which can be re-written as:

$$\begin{aligned}\mathrm{KL}\left[q_\phi(\mathbf{\Omega}) \| p(\mathbf{\Omega})\right] = &\sum_{c=1}^{C}\sum_{k_c=1}^{K_c} \mathrm{KL}\big[q_\phi(\mathcal{W}^{(k_c)}) \| p(\mathcal{W}^{(k_c)})\big] \\ &+ \sum_{l=1}^{L} \mathrm{KL}\big[q_\phi(\mathbf{W}^{(l)}) \| p(\mathbf{W}^{(l)})\big].\end{aligned} \tag{3}$$

B. Tensor Normal Distribution (TND)

We consider convolutional kernels as three-dimensional (3-D) tensors and define tensor normal distributions (TNDs) as prior distributions over these kernels [37]. A TND of order 3 is represented by: $\mathcal{W} \sim \mathcal{TN}_{n_1,n_2,n_3}(\mathcal{M}, \mathcal{T})$, where $\mathcal{M} = E[\mathcal{W}]$ is the mean tensor, and $\mathcal{T}$ is the covariance tensor of order 6 defined as,

$$\mathcal{T}_{i_1 i_2 i_3 j_1 j_2 j_3} = E\big[\mathcal{W}_{i_1 i_2 i_3}\ \mathcal{W}_{j_1 j_2 j_3}\big]. \tag{4}$$

It can be shown that the covariance tensor in (4) is positive semi-definite. In a separable or Kronecker structured model, the covariance matrix of the vectorized multi-dimensional array is the Kronecker product of a number of covariance matrices equal to the number of dimensions, which is 3 in our case, i.e., $\mathcal{T} = \bigotimes_{j=1}^{3} \mathbf{U}^{(j)}$, where $\{\mathbf{U}^{(j)}\}_{j=1}^{3}$ are positive semi-definite matrices [37]. The Kronecker structured model reduces the number of parameters to be estimated.

Proposition 1. *The existence of the factorization, $\mathcal{T} = \bigotimes_{j=3}^{1} \mathbf{U}^{(j)}$, is equivalent to the following analytic condition:*

$$\mathcal{T}_{i_1 i_2 i_3 j_1 j_2 j_3} \mathcal{T}^2_{111111} = \mathcal{T}_{11 i_3 11 j_3} \mathcal{T}_{1 i_2 11 j_2 1} \mathcal{T}_{i_1 11 j_1 11}. \tag{5}$$

The condition in Proposition 1 can be viewed as "non-correlation" between three indices of the tensor $\mathcal{W}_{i_1 i_2 i_3}$. We note from (5) that up to a normalization factor, the covariance coefficient is the product of covariance coefficients for the pairs $i_1 j_1, i_2 j_2, i_3 j_3$ with other indices being fixed equal to 1. In other words, the factorization of the six-dimensional covariance tensor can be interpreted as there is no correlation between the height, width and depth of the 3D kernel. In this case, an equivalent formulation of the TND is a multivariate Gaussian distribution,

$$\mathrm{vec}(\mathcal{W}) \sim \mathcal{N}_{\prod_{j=1}^{3} n_j}(\mathrm{vec}(\mathcal{M}), \bigotimes_{j=3}^{1} \mathbf{U}^{(j)}), \tag{6}$$

where $\mathrm{vec}(.)$ denotes the vectorization operation.

C. Variational Density Propagation

We propose a density propagation framework by propagating the moments of the variational distribution $q_\phi(\mathbf{\Omega})$ through the network's layers and non-linearities. In our settings, the convolutional kernels, the output of activation functions, extracted features, logits, and the soft-max function output are all random variables (as illustrated in Fig. 1). Therefore, instead of performing algebraic operations on real numbers, we are confronted with operations on random variables, including (1) multiplication of a random variable with a constant, (2) multiplication of two random variables, and (3) non-linear transformations operating over random variables. As a result of the multiplication of two Gaussian random variables or the non-linear transformations, the resulting random variables may not have Gaussian distribution [38]. By propagating the mean and covariance of the variation distribution, we obtain the mean and covariance of the predictive distribution, $p(\mathbf{y}|\mathcal{X}, \mathcal{D})$. The mean of $p(\mathbf{y}|\mathcal{X}, \mathcal{D})$ represents the network's prediction, while the covariance matrix reflects the uncertainty associated with the output decision.

D. Extended Variational Density Propagation (exVDP)

In this section, we build the mathematical foundation of the variational density propagation framework by deriving the propagation of the mean and covariance of the variational distribution $q_\phi(\mathbf{\Omega})$ through a convolutional layer, activation function, max-pooling, fully-connected layer, soft-max function, batch normalization and a skip connection mapping. We use the first-order Taylor series for approximating the first two moments (mean and covariance) after a non-linear activation function and refer to this method as the extended variational density propagation (exVDP).

1) First Convolutional Layer: The convolution operation between a set of kernels and the input tensor is formulated as a matrix-vector multiplication. We first form sub-tensors $\mathcal{X}_{i:i+r_1-1, j:j+r_2-1}$ from the input tensor $\mathcal{X}$, having the same size as the kernels $\mathcal{W}^{(k_c)} \in \mathbb{R}^{r_1 \times r_2 \times K}$. These sub-tensors are subsequently vectorized and arranged as the rows of a matrix $\tilde{\mathbf{X}}$. Thus, convolving $\mathcal{X}$ with the k_c^{th} kernel $\mathcal{W}^{(k_c)}$ is equivalent to multiplication of $\tilde{\mathbf{X}}$ with $\mathrm{vec}(\mathcal{W}^{(k_c)})$,

$$\mathbf{z}^{(k_c)} = \mathcal{X} * \mathcal{W}^{(k_c)} = \tilde{\mathbf{X}} \times \mathrm{vec}(\mathcal{W}^{(k_c)}), \tag{7}$$

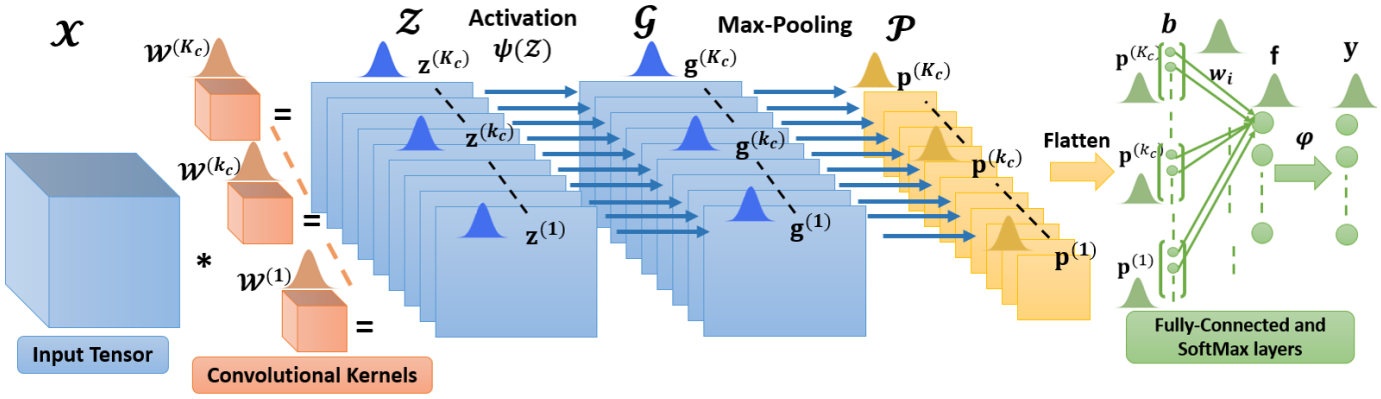

Fig. 1. A schematic layout of the proposed PremiUm-CNN. We show the propagation of the variational density through a convolutional layer, activation function, max-pooling, fully-connected layer and a soft-max function. The convolutional kernels, extracted features, the output of activation functions, logits $(\{\mathcal{W}^{(k_c)}, \mathbf{z}^{(k_c)}, \mathbf{g}^{(k_c)}, \mathbf{p}^{(k_c)}\}_{k_c=1}^{K_c})$, and the soft-max function output $\mathbf{y}$ are all random variables.

where $*$ denotes the convolution operation. We have defined TNDs over kernels, which is equivalent to defining multivariate Gaussian distributions over the vectorized kernels, i.e., $\text{vec}(\mathcal{W}^{(k_c)}) \sim \mathcal{N}\left(\mathbf{m}^{(k_c)}, \boldsymbol{\Sigma}^{(k_c)}\right)$, where $\mathbf{m}^{(k_c)} = \text{vec}(\mathcal{M}^{(k_c)})$ and $\boldsymbol{\Sigma}^{(k_c)} = \mathbf{U}^{(1,k_c)} \otimes \mathbf{U}^{(2,k_c)} \otimes \mathbf{U}^{(3,k_c)}$. It follows that, the output of the convolution is derived by,

$$\mathbf{z}^{(k_s)} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathbf{z}^{(k_c)}} = \tilde{\mathbf{X}}\mathbf{m}^{(k_c)},\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}} = \tilde{\mathbf{X}}\boldsymbol{\Sigma}^{(k_c)}\tilde{\mathbf{X}}^T\right). \quad (8)$$

In the first convolution layer, we assume that the input tensor $\mathcal{X}$ is deterministic for simplicity. Later, in Section III-D7, we generalize the convolution for a random input tensor.

2) Non-linear Activation Function: We approximate the mean and covariance after a non-linear activation function ψ using the first-order Taylor series approximation [38]). Let $\mathbf{g}^{(k_c)} = \psi[\mathbf{z}^{(k_c)}]$, then the mean and covariance of $\mathbf{g}^{(k_c)}$ are derived as follows:

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{g}^{(k_c)}} &\approx \psi(\boldsymbol{\mu}_{\mathbf{z}^{(k_c)}}), \\ \boldsymbol{\Sigma}_{\mathbf{g}^{(k_c)}} &\approx \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}} \odot \left(\nabla\psi(\boldsymbol{\mu}_{\mathbf{z}^{(k_c)}})\ \nabla\psi(\boldsymbol{\mu}_{\mathbf{z}^{(k_c)}})^T\right), \end{aligned} \quad (9)$$

where ∇ is the gradient with respect to $\mathbf{z}^{(k_c)}$ and $\odot$ is the Hadamard product. The state-of-the-art activation functions in DNNs, i.e., the Rectified Linear Unit (ReLU) and its variations can be approximately considered piece-wise linear. Thus, the first-order approximation may provide satisfactory results when propagating the first two moments of the variational distribution through these activation functions.

3) Max-Pooling Layer: For the max-pooling, $\boldsymbol{\mu}_{\mathbf{p}^{(k_c)}} = \text{pool}(\boldsymbol{\mu}_{\mathbf{g}^{(k_c)}})$ and $\boldsymbol{\Sigma}_{\mathbf{p}^{(k_c)}} = \text{co-pool}(\boldsymbol{\Sigma}_{\mathbf{g}^{(k_c)}})$, where pool represents the max-pooling operation on the mean and co-pool represents down-sampling the covariance, i.e., keeping only the rows and columns of $\boldsymbol{\Sigma}_{\mathbf{g}^{(k_c)}}$ that correspond to the pooled mean elements.

4) Flattening Operation: The output tensor $\mathcal{P}$ of the max-pooling layer is vectorized to form the input vector $\mathbf{b}$ of the fully-connected layer (Fig. 1) such that, $\mathbf{b} = \left[\mathbf{p}^{(1)T}, \cdots, \mathbf{p}^{(K_c)T}\right]^T$. The mean and covariance matrix of $\mathbf{b}$ are given by:

$$\boldsymbol{\mu}_{\mathbf{b}} = \begin{bmatrix} \boldsymbol{\mu}_{\mathbf{p}^{(1)}} \\ \vdots \\ \boldsymbol{\mu}_{\mathbf{p}^{(K_c)}} \end{bmatrix}, \boldsymbol{\Sigma}_{\mathbf{b}} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathbf{p}^{(1)}} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \boldsymbol{\Sigma}_{\mathbf{p}^{(K_c)}} \end{bmatrix}. \quad (10)$$

5) Fully-Connected Layer: Let $\mathbf{w}_i \sim \mathcal{N}(\mathbf{m}_i, \boldsymbol{\Sigma}_i)$ be the i^{th} weight vector of the fully-connected layer, where $i = 1, \cdots, H$, and H is the number of output neurons (classes). We note that f_i is the product of two independent random vectors $\mathbf{b}$ and $\mathbf{w}_i$. Let $\mathbf{f}$ be the output vector of the fully-connected layer, then the elements of $\boldsymbol{\mu}_{\mathbf{f}}$ and $\boldsymbol{\Sigma}_{\mathbf{f}}$ are derived by the following proposition,

Proposition 2.

$$\begin{aligned} \mu_{f_i} &= \mathbf{m}_i^T \boldsymbol{\mu}_{\mathbf{b}}, \\ \boldsymbol{\Sigma}_{\mathbf{f}} &= \begin{cases} \text{tr}\left(\boldsymbol{\Sigma}_i\boldsymbol{\Sigma}_{\mathbf{b}}\right) + \mathbf{m}_i^T\boldsymbol{\Sigma}_{\mathbf{b}}\mathbf{m}_j + \boldsymbol{\mu}_{\mathbf{b}}^T\boldsymbol{\Sigma}_i\boldsymbol{\mu}_{\mathbf{b}}, & i = j \\ \mathbf{m}_i^T\boldsymbol{\Sigma}_{\mathbf{b}}\mathbf{m}_j, & i \neq j \end{cases} \end{aligned} \quad (11)$$

where $i, j = 1, \cdots, H$.

6) Soft-max Function: Let the output of the neural network be $\mathbf{y} = \varphi(\mathbf{f})$, where φ is the soft-max function. Using the first-order Taylor series approximation, the mean and covariance of the output vector, i.e., $\boldsymbol{\mu}_{\mathbf{y}}$ and $\boldsymbol{\Sigma}_{\mathbf{y}}$, are derived as follows [20],

$$\boldsymbol{\mu}_{\mathbf{y}} \approx \varphi(\boldsymbol{\mu}_{\mathbf{f}});\ \boldsymbol{\Sigma}_{\mathbf{y}} \approx \mathbf{J}_{\varphi}\boldsymbol{\Sigma}_{\mathbf{f}}\mathbf{J}_{\varphi}^T, \quad (12)$$

where $\mathbf{J}_{\varphi}$ is the Jacobian matrix of φ with respect to $\mathbf{f}$ evaluated at $\boldsymbol{\mu}_{\mathbf{f}}$ [20].

7) Intermediate Convolutional Layers: We generalize the variational density propagation through the convolutional layer for a random input tensor. Consider an intermediate convolutional layer, e.g., the second convolutional layer with a total of K_2 convolution kernels that convolve with the tensor $\mathcal{G}$ (the output tensor of the previous layer, Fig. 2). Similar to the first convolutional layer, we formulate the convolution operation as a matrix-vector multiplication. We form sub-tensors $\mathcal{G}_{i:i+r_1-1,j:j+r_2-1}$ from the tensor $\mathcal{G}$ having the same

size as any kernel $\mathcal{W}^{(k_2)} \in \mathbb{R}^{r_1 \times r_2 \times K_1}$. Every vectorized sub-tensor, say $\mathbf{g}^{(l)} = \text{vec}(\mathcal{G}_{i:i+r_1-1,j:j+r_2-1})$, is multiplied with the vectorized kernel such as,

$$s_l^{(k_2)} = \mathcal{G}_{i:i+r_1-1,j:j+r_2-1} * \mathcal{W}^{(k_c)} = (\mathbf{g}^{(l)})^T \times \text{vec}(\mathcal{W}^{(k_2)}), \tag{13}$$

where $l = 1, \cdots, \dim(\mathbf{s}^{(k_2)})$, $k_2 = 1, \cdots, K_2$ and $\dim(\mathbf{s}^{(k_2)})$ is the dimension of the k_2^{th} slice in the tensor $\mathcal{S}$, (see Fig. 2).

Given that $s_l^{(k_2)}$ is the product of two independent random vectors, $\mathbf{g}^{(l)}$ and $\text{vec}(\mathcal{W}^{(k_2)})$, the mean and covariance matrix of $s_l^{(k_2)}$ can be computed using Proposition 2.

8) Batch Normalization Layer: Batch Normalization (BN) is a differentiable transformation that standardizes the inputs to a layer for each mini-batch [39]. The standardization operation refers to re-scaling layer inputs to have a mean of zero and a standard deviation of one, i.e., standard Gaussian. Consider d-dimensional input vectors of a layer over a mini-batch of size m, i.e., $\mathcal{B} = \{\mathbf{x}_{1,\cdots,m}\}$, where $\mathbf{x}_i \in \mathbb{R}^d$. Then, the output of a batch normalizing transform per i^{th} data sample, $\mathbf{y}_i^{BN}$, (assuming independent data samples), is given by,

$$\mathbf{y}_i^{BN} = \boldsymbol{\gamma} \odot \hat{\mathbf{x}}_i + \boldsymbol{\beta}, \quad \hat{\mathbf{x}}_i = \big(\mathbf{x}_i - \boldsymbol{\mu}_{\mathcal{B}}\big) \odot \frac{1}{\sqrt{\boldsymbol{\sigma}_{\mathcal{B}}^2 + \epsilon}}, \tag{14}$$

where $\boldsymbol{\mu}_{\mathcal{B}}$, $\boldsymbol{\sigma}_{\mathcal{B}}^2$ are the sample mean and sample variance over the mini-batch, $\boldsymbol{\gamma}$, $\boldsymbol{\beta}$ are hyper-parameters for scaling and shifting the input and ϵ is a constant to ensure numerical stability. Since $\boldsymbol{\mu}_{\mathcal{B}}$, $\boldsymbol{\sigma}_{\mathcal{B}}^2$, $\boldsymbol{\gamma}$, $\boldsymbol{\beta}$ and ϵ are deterministic quantities, then the propagation of the first two moments through the BN layer is derived as,

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{y}_i}^{BN} &= \frac{\boldsymbol{\gamma}}{\sqrt{\boldsymbol{\sigma}_{\mathcal{B}}^2 + \epsilon}} \odot (\boldsymbol{\mu}_{\mathbf{x}_i} - \boldsymbol{\mu}_{\mathcal{B}}) + \boldsymbol{\beta}, \\ \boldsymbol{\Sigma}_{\mathbf{y}_i}^{BN} &= Diag\Big(\frac{\boldsymbol{\gamma}}{\sqrt{\boldsymbol{\sigma}_{\mathcal{B}}^2 + \epsilon}}\Big)\, \boldsymbol{\Sigma}_{\mathbf{x}_i} Diag\Big(\frac{\boldsymbol{\gamma}}{\sqrt{\boldsymbol{\sigma}_{\mathcal{B}}^2 + \epsilon}}\Big), \end{aligned} \tag{15}$$

where $Diag(\mathbf{x})$ is a diagonal matrix whose diagonal entries are the entries of the column vector $\mathbf{x}$.

9) Residual Skip Connection Mapping: The skip connection performs an identity mapping that skips two or three layers in a network [40]. The residual block with identity mapping can be mathematically formulated as follows,

$$\mathbf{x}_{l+1} = \mathbf{x}_l + \mathcal{F}(\mathbf{x}_l), \tag{16}$$

where $\mathbf{x}_l$ and $\mathbf{x}_{l+1}$ are the input and output of the l^{th} block in the network and $\mathcal{F}$ is a residual function. The residual function is a non-linear function rendering the output of two or three layers (every layer involves a convolution, BN and a ReLU activation) [40]. Thus, the propagation of the mean and covariance through the skip connection mapping is derived using the first-order Taylor series approximation,

$$\boldsymbol{\mu}_{\mathbf{x}_{l+1}} \approx \boldsymbol{\mu}_{\mathbf{x}_l} + \mathcal{F}(\boldsymbol{\mu}_{\mathbf{x}_l});\ \boldsymbol{\Sigma}_{\mathbf{x}_{l+1}} \approx \mathbf{J}\ \boldsymbol{\Sigma}_{\mathbf{x}_l}\ \mathbf{J}^T, \tag{17}$$

where $\mathbf{J}$ is the Jacobian of $\mathbf{x}_{l+1}$ with respect to $\mathbf{x}_l$. The derivation in Eq. 17 applies to any variation of the residual connection as long as the transformation remains non-linear.

10) Objective Function: Assuming a diagonal covariance matrix for the Initial variational distribution, N independently and identically distributed (iid) data points and using M Monte Carlo samples to approximate the expectation by a summation, the expected log-likelihood in the ELBO objective function is given in Eq. 18. The regularization term in Eq. 3 is the KL-divergence between two multivariate Gaussian distributions and is presented in the Appendix [41].

11) Back-propagation: During back-propagation, we compute the gradient of the objective function, $\nabla_\phi \mathfrak{L}(\phi; \mathcal{D})$, with respect to the variational parameters $\phi = \Big\{\Big\{\big\{\mathfrak{M}^{(k_c)}, \sigma^2_{r_1,k_c}, \sigma^2_{r_2,k_c}, \sigma^2_{K_{c-1},k_c}\big\}_{k_c=1}^{K_c}\Big\}_{c=1}^{C}, \big\{\mathbf{m}_h, \sigma_h^2\big\}_{h=1}^{H}\Big\}$, where $(r_1 \times r_2 \times K_{c-1})$ is the size of the k_c^{th} kernel, K_c is the number of kernels in the c^{th} convolutional layer and H is the number of output neurons. We use $\nabla_\phi \mathfrak{L}(\phi; \mathcal{D})$ to update our parameters ϕ using the gradient descent update rule.

$$E_{q_\phi(\boldsymbol{\Omega})}\{\log p(\mathbf{y}|\mathfrak{X}, \boldsymbol{\Omega})\} \approx -\frac{NH}{2}\log(2\pi) - \frac{1}{M}\sum_{m=1}^{M}\Big[\frac{N}{2}\log(|\boldsymbol{\Sigma}_{\mathbf{y}}|) + \frac{1}{2}\sum_{i=1}^{N}(\mathbf{y}^{(i)} - \boldsymbol{\mu}_{\mathbf{y}}^{(m)})^T(\boldsymbol{\Sigma}_{\mathbf{y}}^{(m)})^{-1}(\mathbf{y}^{(i)} - \boldsymbol{\mu}_{\mathbf{y}}^{(m)})\Big]. \tag{18}$$

E. Unscented Variational Density Propagation (unVDP)

In the exVDP, we have used the first-order Taylor series approximation to estimate the mean and covariance after the non-linear activation functions, which may result in an accumulation of errors, especially in DNNs with a large number of stacked non-linear activation functions. The unscented transformation (UT) approximates the mean and covariance after a non-linear transformation with one or two orders of magnitude better than the first-order approximation in the exVDP model. The UT assures that the estimated mean and covariance are correct, at least up to the third-order for any non-linearity [20]. For non-Gaussian inputs, the UT approximations are accurate at least up to the second-order [22], [23].

In the UT framework, the probability density function (pdf) is specified using a set of carefully chosen samples, called *sigma points*. In Fig. 3, we present an example of using the UT for estimating the mean and covariance of 2D Gaussian distribution passes through a non-linear transformation. Note that the UT approximation differs substantially from general sampling methods (e.g., Monte-Carlo methods), which require orders of magnitude more sample points in an attempt to propagate an accurate (possibly non- Gaussian) distribution of the state. The UT relies on a computationally efficient sampling approach using a finite number of *sigma points*, exactly $(2d)$, with d being the dimension of the random vector. Perhaps, one of the most acclaimed implementations of the UT is the Unscented Kalman Filter (UKF), which recursively estimates the state in non-linear systems [42].

Consider the random vector $\mathbf{z}^{(k_c)}$ of length d, which is transformed by the non-linear activation function ψ, such

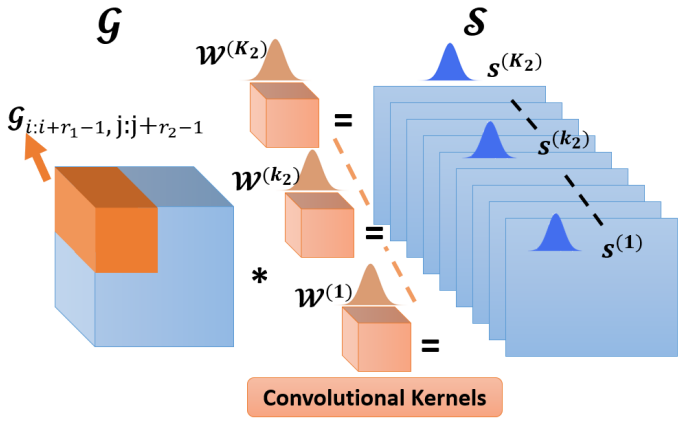

Fig. 2. Propagation of the mean and covariance of the variational distribution $q_\phi(\mathbf{\Omega})$ through the second convolutional layer in a CNN. All elements of $\mathbf{s}^{(k_2)}$ are random variables result from the multiplication of two random vectors, $\text{vec}(\mathcal{G}_{i:i+r_1-1,j:j+r_2-1})$ and $\text{vec}(\mathcal{W}^{(k_2)})$.

that $\mathbf{g}^{(k_c)} = \psi[\mathbf{z}^{(k_c)}]$. The mean vector and the covariance matrix of $\mathbf{z}^{(k_c)}$, i.e., $\boldsymbol{\mu}_{\mathbf{z}^{(k_c)}}$ and $\boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}$ given in Eq. (8), are used to generate $2d$ *sigma points* (Eq. 19), where $\sqrt{d\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}}$ represents the matrix square root such that $\left(\sqrt{d\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}}\right)^T\left(\sqrt{d\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}}\right) = d\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}$, and $\left(\sqrt{d\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}}\right)_i$ is the i^{th} row of $\left(\sqrt{d\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}}\right)$.

$$\begin{aligned}
\mathbf{z}^{(k_c,\ i)} &= \boldsymbol{\mu}_{\mathbf{z}^{(k_c)}} + \tilde{\mathbf{z}}^{(k_c,\ i)}, && i = 1, \cdots, 2d \\
\tilde{\mathbf{z}}^{(k_c,\ i)} &= \left(\sqrt{d\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}}\right)_i^T, && i = 1, \cdots, d \\
\tilde{\mathbf{z}}^{(k_c,\ d+i)} &= -\left(\sqrt{d\ \boldsymbol{\Sigma}_{\mathbf{z}^{(k_c)}}}\right)_i^T, && i = 1, \cdots, d.
\end{aligned} \tag{19}$$

Each individual *sigma point* is transformed using the non-linear activation function $\psi(.)$, i.e., $\mathbf{g}^{(k_c,\ i)} = \psi[\mathbf{z}^{(k_c,\ i)}]$, where $i = 1, \cdots, 2d$. The approximate mean and covariance of $\mathbf{g}^{(k_c)}$ are then computed as follows,

$$\begin{aligned}
\boldsymbol{\mu}_{\mathbf{g}^{(k_c)}} &= \frac{1}{2d}\sum_{i=1}^{2d}\mathbf{g}^{(k_c,i)} \\
\boldsymbol{\Sigma}_{\mathbf{g}^{(k_c)}} &= \frac{1}{2d}\sum_{i=1}^{2d}(\mathbf{g}^{(k_c,i)} - \boldsymbol{\mu}_{\mathbf{g}^{(k_c)}})(\mathbf{g}^{(k_c,i)} - \boldsymbol{\mu}_{\mathbf{g}^{(k_c)}})^T.
\end{aligned} \tag{20}$$

F. Computational Complexity

The computational complexity of the proposed PremiUm-CNN models, i.e., exVDP and unVDP, is comparable to the computational complexity of their deterministic homologs. Since we have assumed a diagonal covariance matrix for the initial variational distribution, the TND, defined in Eq. 6, adds a single additional parameter (i.e., the variance) for each convolutional kernel, and a single additional parameter for each weight vector $\mathbf{w}_i$ in the fully-connected layer (Section III-D5). The mean-covariance propagation through the non-linearities is performed with NO additional parameters using the first-order Taylor series (exVDP) or the unscented transformation (unVDP). For instance, a CNN with $C = 6$ convolutional layers ($K_c = 32$ kernels of size 5×5 in each layer) and $L = 1$ fully-connected layer ($H = 10$ the size of the output vector) has a total of 32*5*5*6 + 20,480 =25,280 parameters. Thus, the proposed density propagation models result in an increase of 32 parameters in each convolutional layer and 10 additional parameters in the fully connected layer. The total increase in the number of parameters (i.e., $32*6+10 = 202$) is, therefore, negligible. Moreover, the computational cost of the UT is proportional to the number of *sigma points*, $2d$, where d, in our case, is the size of the feature map after a convolutional layer. For example, in the MNIST dataset, the feature map after a convolution layer with a single kernel is 24×24. Hence, the number of *sigma points* is $2 * 24 * 24 = 1,152$ (assuming no padding).

IV. Experiments and Results

In this section, we assess the performance of the proposed exVDP and unVDP approaches for classification and segmentation tasks. For the classification task, we use two benchmark datasets, i.e., MNIST and CIFAR-10 [43], [44]. We use synthetic aperture radar (SAR) images and magnetic resonance (MR) images of the brain tumor for the segmentation task [45], [46]. We compare our results with the state-of-the-art, including Bayes-by-Backprop (BBB), Bayes-CNN, Dropout CNN, and a deterministic CNN (deterministic means without uncertainty) [11], [17], [19]. The classification or segmentation accuracy is used as a metric to evaluate the performance on the test datasets. Adam algorithm is used as an optimizer in all experiments [47].

We establish the robustness of the proposed models against various levels of Gaussian noise and adversarial attacks. The targeted adversarial examples were generated using the fast gradient sign method (FGSM) and projected gradient descent (PGD) [48], [49]. We used three noise levels, i.e., low, medium and high, based on the amount of noise required to introduce significant distortion in the input test dataset. For MNIST classification task, the noise levels were $\sigma_{\text{noise}} = 0.1, 0.2, 0.3$ for adversarial noise and $\sigma^2_{\text{noise}} = 0.1, 0.2, 0.3$ for Gaussian noise. For CIFAR-10 classification, the three noise levels were measured by the highest conceivable value (HCV= $0.01, 0.05, 0.1$) for both Gaussian and adversarial noise, where HCV $= 3\ \sigma_{\text{noise}}$ [50]. Similarly, in the Flevoland SAR segmentation task, the noise levels were HCV= $0.1, 0.2, 0.3$

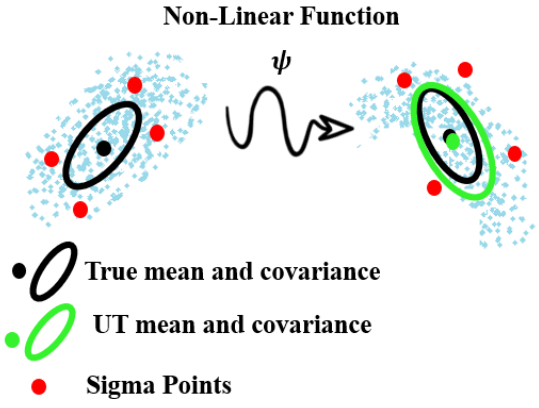

Fig. 3. A schematic description of the unscented transformation (UT). We approximate the mean and covariance of a 2D Gaussian distribution after a non-linear function ψ using 4 *sigma points* (computed by Eqs. 19 and 20).

for adversarial noise and $\sigma_{\text{noise}} = 0.1, 0.2, 0.3$ for Gaussian noise. In the Oberpfaffenhofen SAR segmentation, adversarial noise is measured by HCV$= 0.01, 0.05, 0.1$. For the MRI segmentation task, only a medium level of noise (HCV$= 0.05$) was used for Gaussian and adversarial noise. All experiments are done using two Lambda servers with 16 RTX-600 GPUs available at Rowan University AI Lab[1].

A. Image Classification - MNIST and CIFAR-10

The proposed model architecture for the MNIST dataset is one convolutional layer (32 kernels each of size 5×5) followed by a rectified linear unit (ReLU) activation, one max-pooling, and one fully-connected layer. We set the learning rate to 0.0001, the batch size to 50 and the number of epochs to 4. In Table I, we present the test accuracy of the unVDP, exVDP, BBB, and a CNN at three levels of FGSM adversarial and Gaussian noise. The targeted adversarial examples are generated to fool each network into predicting digit "3". We note the performance decline of all networks when Gaussian or adversarial noise is added; however, unVDP and exVDP models are significantly more robust (higher test accuracy).

For the CIFAR-10 dataset, the CNN architecture is six convolutional layers, each followed by an exponential linear unit (ELU) activation, three max-pooling, and one fully-connected layer [51]. The numbers of convolutional kernels in layers one to six are $32, 32, 64, 64, 128$ and 128, respectively. We use kernels of size (3×3) in all layers. We set the learning rate set to 10^{-5}, the batch size to 50 and the number of epochs to 250.

In Table II, we present the test accuracy of the unVDP, exVDP, Bayes-CNN, and Dropout CNN with varying levels of FGSM adversarial and Gaussian noise. The targeted class is selected as a "cat". We note that all networks perform well on noise-free test data; however, it is evident that unVDP and exVDP maintain their classification performance under adversarial attacks and Gaussian noise. In Tables I and II, we present inference time per data sample (image) in the last row to understand the trade-off for the robustness.

Our experiments also include the CNN architecture with 10 convolutional, 1 fully-connected and 5 max-poling layers. We add batch normalization after every ELU activation function, with hyperparameters set as $\boldsymbol{\gamma} = 1$, $\boldsymbol{\beta} = 0$ and $\epsilon = 10^{-4}$ (Eq. 14). The numbers of convolutional kernels in layers one to ten are $32, 32, 32, 32, 64, 64, 64, 128, 128$ and 128, respectively. The size of the kernels is 3×3 in all layers except the first one (which has 5×5) and the last one, which has 1×1. We use a learning rate scheduler with polynomial decay and set the initial learning rate to 10^{-3}. We evaluate the performance of the proposed variational density propagation framework for a residual neural network (referred to as VDP-ResNet) with 18 layers (Eq. 17) against targeted FGSM and PGD adversarial attacks (white-box and black-box attacks). In the implementation of the skip-connection, we approximate the Jacobian matrix (Eq. 17) required for the propagation of the variance-covariance matrix in the forward pass by an identity matrix. We set the magnitude of the adversarial noise to $HCV/3$, the step size in the PGD attack to 1 and the maximum number of iterations to 40. The subscript B in the models unVDP$_B$, exVDP$_B$ and VDP-ResNet$_B$ refers to the black-box attack. The black-box attack is generated from the same model architecture trained with the same training specifics but without propagating variational distribution. Thus, the attack does not have access to the model gradient. Table III shows the test accuracy of the unVDP, exVDP, Dropout-CNN, VDP-ResNet, and ResNet for the CIFAR-10 dataset. We note from Table III that the VDP models retain high accuracy under adversarial attacks, which demonstrates the robustness of the proposed density propagation framework for deeper network architectures. We also note that VDP models present higher accuracy under the black-box attacks as compared to white-box attacks.

[1]Source code available at https://bit.ly/3ifCTgW

TABLE I
MNIST TEST ACCURACY AT VARYING LEVELS OF FGSM ADVERSARIAL AND GAUSSIAN NOISE

Gaussian noise level	unVDP	exVDP	BBB	CNN
No noise	97.9%	97.8%	97.8%	97.7%
Low	95.1%	94.1%	86.4%	79.6%
Medium	86.7%	84.6%	76.7%	70.5%
High	74.8%	73.4%	63.8%	55.9%
Adversarial noise level				
Low	97.5%	96.6%	91.5%	58.7%
Medium	84.9%	84.4%	45.9%	14.7%
High	66.1%	51.6%	16.5%	14.5%
Inference time per data sample (sec $*10^{-2}$)	2.6	1.2	.03	.02

B. Image Segmentation - SAR Image Datasets

Two publicly available SAR datasets are used to evaluate the performance, (1) Airborne SAR (AIRSAR) data of agricultural area over Flevoland in The Netherlands, and (2) the electronically steered array radar (ESAR) data collected over Oberpfaffenhofen, Germany [45]. We set up the segmentation task as a

TABLE II
CIFAR-10 TEST ACCURACY AT VARYING LEVELS OF FGSM ADVERSARIAL AND GAUSSIAN NOISE

Gaussian noise level	*unVDP*	*exVDP*	Bayes-CNN	Dropout CNN
No noise	92.5%	91.8%	92.1%	91.5%
Low	92.3%	91.4%	87.0%	89.0%
Medium	91.9%	90.9%	86.8%	87.2%
High	90.1%	89.1%	85.2%	86.0%
Adversarial noise level				
Low	88.2%	88.1%	76.2%	77.0%
Medium	85.4%	82.3%	69.1%	53.0%
High	76.5%	67.7%	42.2%	33.0%
Inference time per data sample (sec $*10^{-2}$)	11.5	5.	1.	.8

TABLE III
CIFAR-10 TEST ACCURACY USING DEEP CNN AND RESNET ARCHITECTURES AT VARYING LEVELS OF WHITE AND BLACK FGSM AND PGD ADVERSARIAL ATTACKS

	Noise level	*unVDP*	*unVDP*$_B$	*exVDP*	*exVDP*$_B$	Dropout-CNN	VDP-ResNet	VDP-ResNet$_B$	ResNet
	No noise	91.7%	-	91.8%	-	91.7%	90.0%	-	91.1%
FGSM	Low	91.2%	91.5%	91.2%	91.4%	86.8%	89.2%	89.4%	82.4%
	Medium	83.4%	91.4%	83.8%	91.2%	65.9%	83.6%	86.2%	56.1%
	High	71.1%	88.8%	70.6%	89.1%	56.9%	79.1%	79.0%	47.5%
PGD	Low	91.1%	91.4%	91.2%	91.4%	85.5%	88.6%	89.8%	75.1%
	Medium	82.8%	91.1%	82.2%	91.0%	54.7%	78.4%	85.5%	23.3%
	High	70.5%	88.6%	69.7%	88.5%	42.9%	69.8%	79.3%	13.8%

region-based classification problem following the work of [52]. We start with randomly sampling patches of size 8×8 from SAR images, which are used as inputs to the networks [53]. The label of each patch is set to the label of the center pixel of the patch. We compare the proposed exVDP and unVDP models with a CNN in [52]. The following architecture is used for all three networks: two convolutional layers, two ReLU layers, one max-pooling, one fully connected, and one soft-max layer. The first convolutional layer has 64 kernels of size $3\times3\times6$, and the second layer has 128 kernels of size $2\times2\times64$. We evaluate the performance of all three networks at various levels of Gaussian noise and FGSM adversarial attacks. At test time, the output covariance matrices (for all patches) are used to generate *uncertainty maps*, which represent pixel-level confidence in the segmentation output.

1) Flevoland Dataset: Flevoland image is a subset of an L-band, full SAR image, acquired by the NASA/Jet Propulsion Laboratory AIRSAR platform in 1989 during the MAESTRO-1 Campaign [45]. The size of the SAR image is 750×1024 pixels$\times 6$ channels and it contains 15 object classes [45]. We sample $30,000$ patches from the image and divide these patches into training and validation bins (90% for training and 10% for validation). The test accuracy is computed using $156,741$ patches sampled from the entire image [53]. We set "lucerne" as the target class for the adversarial attacks. We set the learning rate to 10^{-4} for exVDP and 10^{-5} for unVDP, the batch size to 50 and the number of epochs to 120.

We presents the segmentation accuracy in Table IV for three models, i.e., unVDP, exVDP and the deterministic CNN in [52] for varying levels of Gaussian and adversarial noise. We note that all three models perform well on the noise-free case; however, exVDP and unVDP maintain significantly higher accuracy in the case of Gaussian noise or adversarial attacks.

In Fig. 4, we present the Flevoland SAR image with the ground truth segmentation annotations and the predicted segmentation by unVDP model in four cases: the noise-free case and three levels of adversarial noise. We also present the uncertainty maps for the noisy cases, which represent the model's confidence in segmentation decisions. We observe that the uncertainty in the segmentation results increases as the noise level increases. The uncertainty maps show darker regions indicated by arrows at higher noise levels.

TABLE IV
FLEVOLAND SAR IMAGE - SEGMENTATION ACCURACY FOR VARIOUS GAUSSIAN AND FGSM ADVERSARIAL NOISE LEVELS

Gaussian noise level	*unVDP*	*exVDP*	CNN
No noise	96.7%	96.5%	96.2%
Low	92.3%	90.8%	88.1 %
Medium	85.1%	83.9%	77.9%
High	78.1%	77.8%	69.9 %
Adversarial noise level			
Low	86.5%	83.6%	73.6%
Medium	73.7%	68.8%	56.8%
High	61.8%	56.1%	47.4 %
Inference time per patch sample (sec $*10^{-2}$)	2.8	1.9	1.

2) Oberpfaffenhofen Dataset: The Oberpfaffenhofen SAR image is of size 1300×1200 pixels with 6 channels and has three object classes. We sample $100,000$ patches and divide them into training and testing bins (95% for training and 5% for validation). We use $1,303,960$ patches sampled from the entire SAR image for testing. The target class for the adversarial attack is the "open areas". We set the learning rate to 10^{-4}, the batch size to 100 and the number of epochs to 100. In Fig. 5, we present Oberpfaffenhofen SAR image with (b) the ground truth segmentation, (c) the predicted segmentation by unVDP model in the noise-free case and (d-f) in three levels of adversarial attack. The uncertainty maps are provided in the third row of Fig. 5 for three levels of adversarial noise. We notice that pixels' magnitude is getting higher in the uncertainty maps as the noise increases. Thus, the model is less confident (provides higher uncertainty levels) when the noise level becomes higher.

In Table V, we present the segmentation accuracy of the unVDP, exVDP and the CNN in [52] for the noise-free case and three levels of FGSM adversarial noise. The proposed models resist adversarial attacks and maintain a higher accuracy as compared to their counterpart CNN. We notice that unVDP is more robust than exVDP. The higher accuracy is linked to the UT in the unVDP model that better approximates variational density as compared to the first-order approximation in the exVDP model. The predicted segmentation and uncertainty

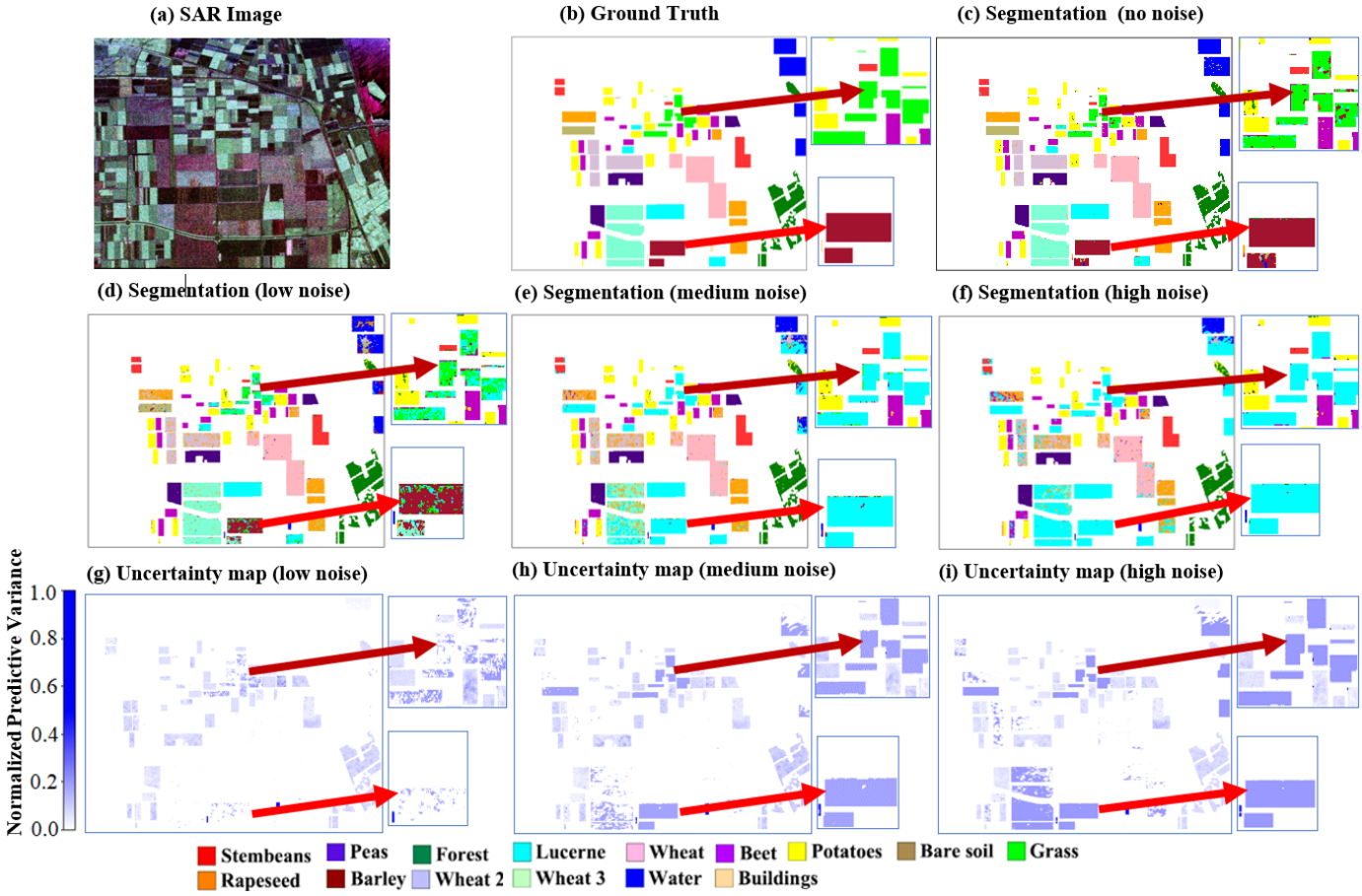

Fig. 4. Flevoland dataset. (a) The Flevoland SAR image. (b) The ground truth segmentation. (c - f) The predicted segmentation by unVDP model for different levels of adversarial noise. (g - i) The uncertainty maps of the unVDP model. The class label "lucerne" (cyan color) is the target of the attack. The arrows refer to pixels that are misclassified as "lucerne" and to the uncertainty associated with those pixels in uncertainty maps.

maps of the exVDP model on Flevoland and Oberpfaffenhofen SAR images for the different levels of adversarial noise are provided in the supplementary materials.

C. Image Segmentation - Brain Tumor Scans

The segmentation of tumors in brain MR images is a challenging problem since the shape, structure, and location of brain tumors are highly variable, especially high grade-gliomas (HGG) [54]. We evaluate the performance of the proposed exVDP and unVDP models on HGG brain tumor segmentation task using Brain Tumor Segmentation Challenge (BraTS) 2015 dataset. The dataset consists of 5 classes, i.e. class 0 - normal tissue, class 1 - necrosis, class 2 - edema, class 3 - non-enhancing, and class 4 - enhancing tumor [46]. The evaluation of the segmentation is based on three regions, (1) complete tumor (classes 1, 2, 3 and 4), (2) tumor core (classes 1, 3 and 4), and (3) enhancing tumor (class 4) [46]. We formulate brain tumor segmentation as a multi-class classification problem by randomly sampling patches from four MR modalities, i.e., FLAIR, T1, T2 and T1C [55]. We manually set the label of each patch to the label of the center pixel. We sample a total of $100,000$ patches of size 33×33 from MR data of 20 patients and divide these patches into training and validation bins (95% for training and 5% for validation). Our test set includes randomly sampled 372 images, i.e., $43,264$ patches, from each of the four modalities. We compare the proposed exVDP and unVDP models with a CNN as proposed in [55]. We use a CNN architecture with six convolution layers (all kernels have a size of 3×3, and we have 32, 32, 64, 64, 128, 128 kernels in layers one to six, respectively, followed by ReLU activation), two max-pooling layers, and a fully-connected layer. The evaluation metric in brain segmentation is Dice Similarity Coefficient (DSC). We evaluate the models before and after adding Gaussian noise or targeted FGSM adversarial attack (targeted class is class 3, i.e., "non-enhancing tumor").

TABLE V
OBERPFAFFENHOFEN SAR IMAGE - SEGMENTATION ACCURACY FOR VARIOUS LEVELS OF FGSM ADVERSARIAL NOISE

Adversarial noise level	*unVDP*	*exVDP*	CNN
No noise	94.2%	94.2%	94.5%
Low	91.4%	89.7%	89.5%
Medium	79.2%	74.1%	64.2%
High	68.7%	67.7%	59.3 %
Inference time per patch sample (sec $*10^{-2}$)	2.5	1.8	.3

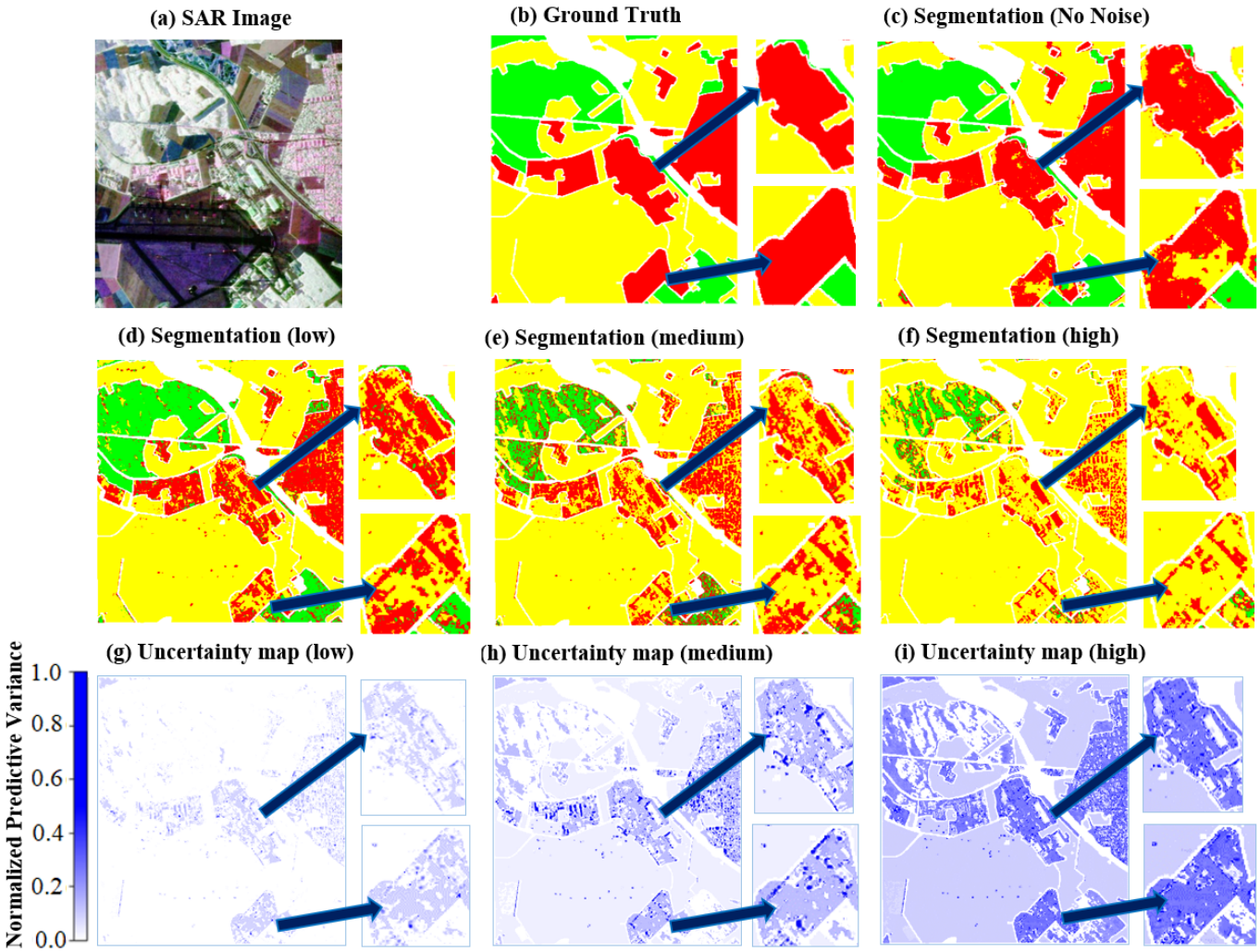

Fig. 5. Oberpfaffenhofen SAR dataset. (a) The original SAR image. (b) The ground truth segmentation. (c - f) The predicted segmentation of the proposed unVDP for varying levels of adversarial noise. (g - i) The uncertainty maps of the unVDP model. The class label "open areas" (yellow color) is the target of the attack. The arrows refer to the pixels that are misclassified as "open areas" and to the uncertainty associated with those pixels in the uncertainty maps.

We set the learning rate to 10^{-4}, the batch size to 50 and the number of epochs to 200.

In Table VI, we present DSC values for three test cases, i.e., noise-free, Gaussian, and adversarial noise. We note that the DSC values of the proposed models are significantly higher than that of the counterpart CNN for all cases in general and adversarial noise in particular. In Fig. 6, we show the predicted segmentation for the exVDP, unVDP and the classic CNN in [55] for a representative HGG image (with and without adversarial noise). The uncertainty maps associated with each segmentation are also presented for both models. The uncertainty maps show higher levels of uncertainty on the tumor boundaries. This allows physicians to quickly review the segmentation results and, if needed, make corrections of tumor boundaries in the regions where the uncertainty is high.

TABLE VI
SEGMENTATION RESULTS MEASURED BY THE DICE SIMILARITY COEFFICIENT (DSC) FOR THE BRATS TEST DATASET.

Method	Tumor Regions	Noise free	Adversarial	Gaussian
unVDP	Complete	**.853**	**.817**	**.831**
	Core	.819	.787	.807
	Enhancing	.837	.754	.817
exVDP	Complete	**.808**	**.774**	**.806**
	Core	.746	.726	.745
	Enhancing	.742	.698	.739
CNN	Complete	.781	.434	.669
	Core	.651	.471	.519
	Enhancing	.752	.439	.557

V. SELF-AWARENESS AND ROBUSTNESS

At the output of the proposed PremiUm-CNN models, i.e., exVDP and unVDP, the mean vector after the soft-max function provides the prediction and the covariance matrix captures the uncertainty in the output decisions. The proposed

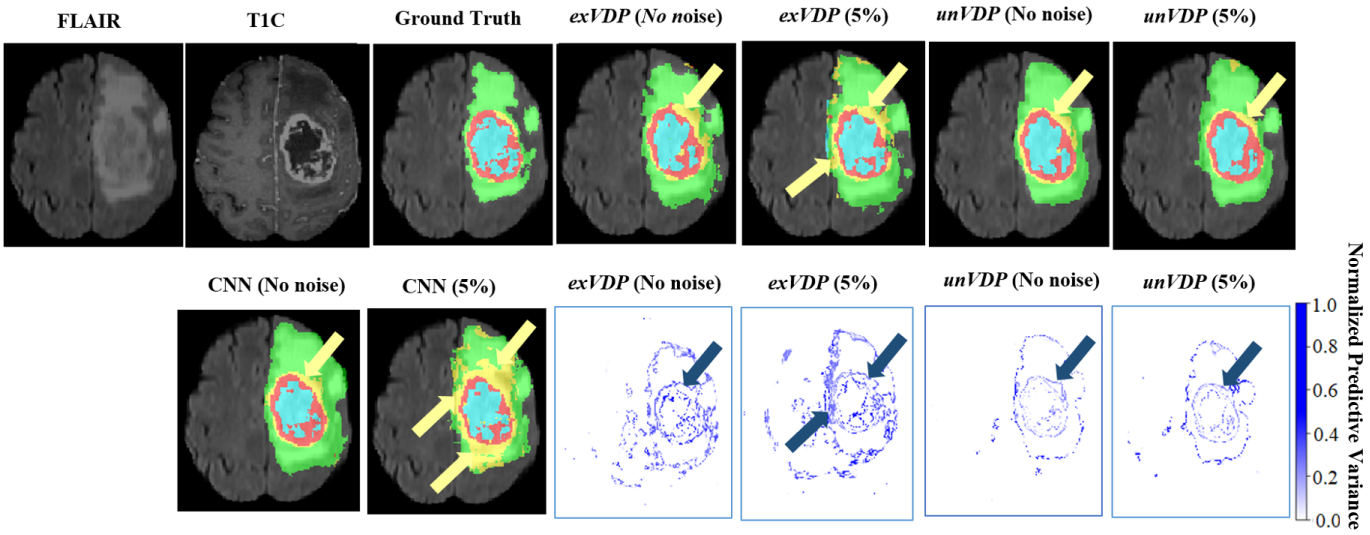

Fig. 6. The predicted Segmentation by the proposed exVDP, unVDP and deterministic CNN on one example of HGG tumor from the BraTS dataset with and without adding adversarial noise. The uncertainty maps associated with each segmentation is also shown. The class label "non-enhancing tumor" (yellow color) is the target of the attack. The green color refers to the edema, the red color refers to the enhancing tumor and the blue color refers to the necrosis.

models can use their confidence/uncertainty information as a quantitative metric to assess their own performance, i.e., leading to models that are "self-aware". This section provides a detailed analysis of the output covariance matrix for both the exVDP and unVDP models linked to the output decisions. Later, we show that propagating uncertainty in the exVDP or unVDP models naturally leads to increased robustness through "logit squeezing".

A. Analysis of the Output Covariance Matrix

We analyze the output covariance matrices of both the exVDP and unVDP models at various levels of Gaussian and adversarial noise for MNIST and CIFAR-10 test datasets. Our analysis reveals that the output covariance matrix for any test example consists of elements with a very small magnitude. However, as we introduce noise, the variance values (diagonal elements of the covariance matrix) and the corresponding covariance elements (off-diagonal elements corresponding to the predicted class) start increasing in magnitude.

1) Output Variance: We compute the average variance over all test examples at various Gaussian and adversarial noise levels, as demonstrated in Figs. 7 (a and b) for MNIST dataset. The same noise levels are used for both Gaussian and adversarial noise and plotted as the signal-to-noise ratio (SNR) in Fig. 7. We also show the test accuracy corresponding to each noise level for both the exVDP and unVDP models in Fig. 7 (c and d).

We observe high average predictive variance ($\geq$ 5) at SNR $\leq$ 7.5 dB for adversarial noise (for both exVDP and unVDP). At high SNR values (exVDP $\sim$12 dB for Gaussian noise and $\sim$20 dB for the adversarial noise, unVDP $\sim$12 dB for both Gaussian and adversarial noise), the average predictive variance settles around 2.4 for the exVDP and 1.8 for the unVDP. We also note that the rate of increase in the average predictive variance is faster for the adversarial noise as compared to Gaussian noise for decreasing SNR (from right to left in Figs. 7 (a and b)). The average predictive variance available at the output of the proposed exVDP and unVDP models can be used at the prediction / inference time to identify the increasing noise in the input or possible adversarial attack. This is referred to as self-awareness for deep neural networks. We provide a similar analysis for the CIFAR-10 dataset in the Appendix.

2) Output Covariance: We also investigate the off-diagonal elements of the output covariance matrix, which represent the covariance values between different output classes. We observe that the magnitudes of the off-diagonal elements increase with increasing noise (or decreasing SNR). In Fig. 8, we present heat-maps of the normalized average of the output covariance matrices of the exVDP and unVDP models on the MNIST dataset for three cases, i.e., noise-free, Gaussian noise, and adversarial noise. The average test accuracies of the unVDP for the three cases are 97.9%, 86.7%, and 84.9%, respectively. Each pixel of the heat-map is a normalized average of the absolute value of the covariance for all 10,000 test examples. The targeted adversarial examples are generated to fool the model into predicting digit "3". For the noise-free case, we observe that the off-diagonal elements have small (close to zero) magnitude (Fig. 8 (a)). For the Gaussian noise (Fig. 8 (b)), we notice that the covariance values are higher; however, there is no pattern, and high and low values are randomly distributed among various classes. Finally, for the targeted adversarial attack (target was "3", Fig. 8 (c)), we observe a high covariance value between the targeted class and all other classes. This pattern of increased covariance values clearly indicates that the model is under a targeted adversarial attack.

B. Robustness through Logit Squeezing

Logit squeezing refers to the technique that penalizes the norm of logits (where logits are class scores that are not

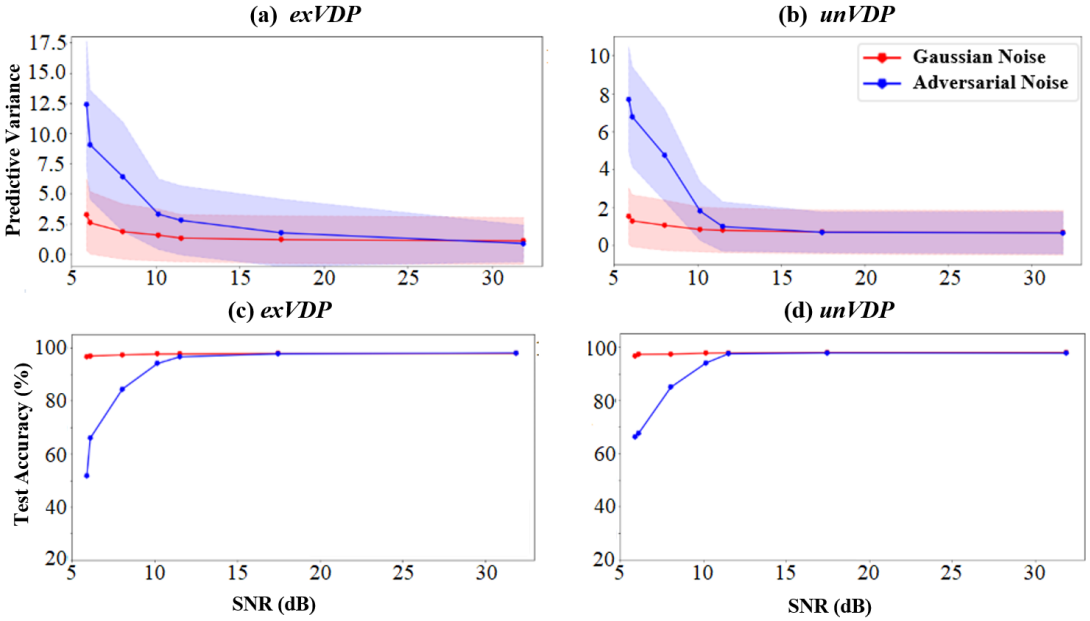

Fig. 7. (a) and (b) Average variance values are plotted against the signal-to-noise ratio (SNR) for varying levels of Gaussian and adversarial noise for exVDP and unVDP, respectively. The variance values are averaged over all 10,000 test examples of the MNIST dataset and the lightly filled areas represent the standard deviation. (c) and (d) Average test accuracy correspond to different noise levels is presented for exVDP and unVDP, respectively.

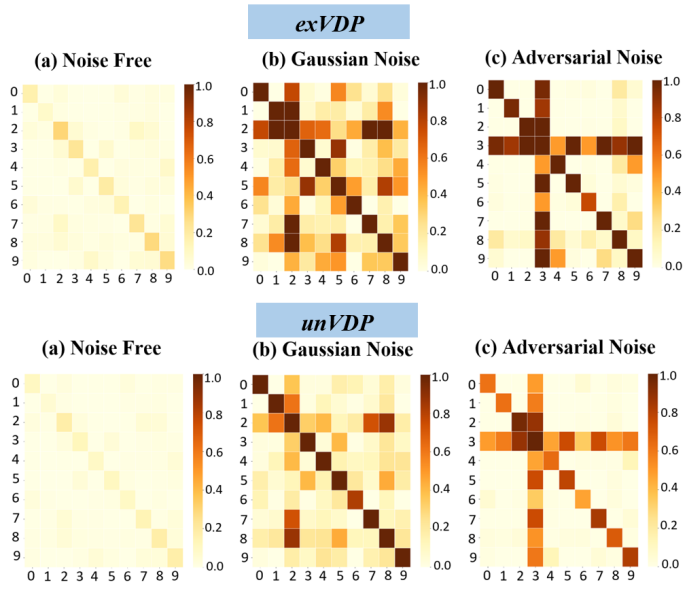

Fig. 8. The heat-maps representing the average of the covariance matrices at the output of the proposed exVDP and unVDP models. Each pixel of the heat-map is a normalized average of the absolute value of the covariance for all 10,000 MNIST test examples. (a) noise-free, (b) Gaussian noise, and (c) adversarial noise. The adversarial examples were generated to fool the models into predicting and image as digit "3".

normalized and form the input to the soft-max function) [56]. In the literature, it was shown that adding a regularization term to the training objective function can explicitly penalize large logits [56], [57]. Shafahi *et al.* have shown that aggressive logit squeezing, by highly weighting the logits regularization term, results in a model that is more robust than a model trained with adversarial examples [57].

We hypothesize that the increased robustness of the proposed density propagation approaches, i.e., exVDP and unVDP is linked to the logit squeezing technique. Therefore, we investigate how the logits change for the exVDP and unVDP compared with a deterministic CNN at varying levels of adversarial noise. In Fig. 9, we present histograms of the logits that correspond to the predicted classes in the MNIST test set for three models, i.e., unVDP, exVDP and a deterministic CNN tested under four adversarial noise conditions (no noise, low noise, medium noise and high noise). We notice that with an increasing level of adversarial noise, the distributions of the logits for the exVDP and unVDP models are squeezed toward lower values. We conjecture that the proposed exVDP and unVDP inherently squeeze the logits without adding any explicit penalty to the objective function (Eq. 2) for the logit squeezing. We believe that the logit squeezing in the exVDP and unVDP may be linked to the second-moment propagation through layers of the DNN. We consider that during training, the availability of additional information in the form of the second moment (variance-covariance matrix) helps the learning process, and results in the proposed models that are more robust to noise and adversarial attacks. However, it has been shown later in the literature that logit squeezing may not always capture the true robustness of models against adversarial attacks [58]. Logit squeezing and the related concept of label smoothing may distort the loss surface in the data space and lead a white-box attacker to exploit useless gradients, i.e., a phenomenon known as gradient masking. Gradient masking leads to a false sense of adversarial robustness as black-box attackers may never need the gradient information to generate adversarial attacks [58], [59].

One sanity check method to verify the existence of gradient masking is to ascertain the performance of the pro-

posed models under black-box attacks in addition to white-box attacks and compare these with each other. When the model accuracy values for the black-box attacks are lower than that of the white-box attacks, this indicates that the proposed models may have the gradient masking problem. In this setting, the black-box attackers are more successful as they do not need true gradient information for generating black-box adversarial attacks [59]. On the other hand, if the models maintain comparatively high accuracy under black-box attacks as compared to white-box attacks, this establishes that the proposed models do not implement gradient masking. We verify that the proposed density propagation framework does not have the gradient masking problem by evaluating our models under black-box attacks and comparing these with white-box attacks. The results are presented in Table III. We notice that the proposed models maintain higher accuracy under the FGSM and PGD black-box attacks as compared to the white-box attacks. Thus, the proposed density propagation framework does not suffer from the gradient masking problem.

VI. Conclusion

In this paper, we propose a framework for Propagating Uncertainty in Convolution Neural Networks, PremiUm-CNNs, which enables the estimation of uncertainty at the output decision. Given that the true density propagation is intractable in DNNs, we propose two methods for approximating variational density, i.e., exVDP and unVDP. In the exVDP model, we have used the first-order approximation for propagating the first two moments of the variational distribution through layers of a CNN. While in the unVDP model, we have used unscented transformation to propagate a set of *sigma points*, which approximate variational distribution at least up to the second order. Our proposed models have naturally led to the logit squeezing mechanism and have significantly enhanced the robustness against noise and adversarial attacks. The experimental results on MNIST, CIFAR-10, SAR images and BraTS 2015 datasets have established superior robustness to Gaussian noise and adversarial attacks as compared to the deterministic CNNs and state-of-the-art Bayesian networks. We have also shown that density propagation in DNNs inherently results in self-aware models that can assess their own performance and detect targeted adversarial attacks.

APPENDIX

A. Regularization Term in ELBO Objective Function

If we have a CNN with one convolutional layer followed by the activation function, one max-pooling and one fully-connected layer, thus the regularization term in the ELBO objective function is derived in Eq. 21, where $(r_1 \times r_2 \times K)$ is the size of the kernels, K_1 is the number of kernels in the first convolutional layer, H is the number of output neurons and n is length of the weight vector $\mathbf{w}_i$ in the fully-connected layer.

$$\begin{aligned}\mathrm{KL}\big[q_\phi(\mathbf{\Omega})\|p(\mathbf{\Omega})\big] = \frac{1}{2}\sum_{k=1}^{K_1}\Big(\|\mathbf{M}^{(k)}\|_F^2 - r_1 r_2 K\,\Big(1-\sigma_{r_1,k}^2\,\sigma_{r_2,k}^2\,\sigma_{K,k}^2+\log\sigma_{r_1,k}^2+\log\sigma_{r_2,k}^2+\log\sigma_{K,k}^2\Big)\Big)\\ +\frac{1}{2}\sum_{i=1}^{H}\Big(\|\mathbf{m}_i\|_F^2 - n\,\big(1-\sigma_i^2+\log\sigma_i^2\big)\Big),\end{aligned} \tag{21}$$

B. Proof of Proposition 1

Consider a covariance tensor of order six, i.e. $\mathcal{T}$ defined as,

$$\mathcal{T}_{i_1i_2i_3j_1j_2j_3} = E\big(\mathcal{W}_{i_1i_2i_3}\;\mathcal{W}_{j_1j_2j_3}\big). \tag{22}$$

The tensor $\mathcal{T}$ is called positive-definite if for any third-order tensor $\mathcal{Y}$, we have (notice that we consider only real-valued tensors):

$$\sum_{i_1i_2i_3}\sum_{j_1j_2j_3}\mathcal{T}_{i_1i_2i_3j_1j_2j_3}\;\mathcal{Y}_{i_1i_2i_3}\;\mathcal{Y}_{j_1j_2j_3}\geq c\|\mathcal{Y}\|^2,\;\; c>0. \tag{23}$$

Tensor $\mathcal{T}$ is also assumed to be symmetric in the following sense: $\mathcal{T}_{i_1i_2i_3j_1j_2j_3} = \mathcal{T}_{j_1j_2j_3i_1i_2i_3}$. Then, like for matrices, there exists n^3 positive eigenvalues $\boldsymbol{\lambda}_s$ and corresponding eigen-third-order-tensor $\mathcal{Y}^{(s)}$, such that the tensor $\mathcal{T}$ has the form,

$$\mathcal{T} = \sum_{s=1}^{n^3}\boldsymbol{\lambda}_s\mathcal{Y}^{(s)}\otimes\mathcal{Y}^{(s)},\;\text{ i.e.,} \tag{24}$$

$$\mathcal{T}_{i_1i_2i_3j_1j_2j_3} = \sum_{s=1}^{n^3}\boldsymbol{\lambda}_s\mathcal{Y}^{(s)}_{i_1i_2i_3}\mathcal{Y}^{(s)}_{j_1j_2j_3}. \tag{25}$$

Here n is the maximal value for indices $i_1, i_2, i_3, j_1, j_2, j_3$.

Now, we assume that the tensor $\mathcal{T}$ has a special factorized form, i.e.,

$$\mathcal{T}_{i_1i_2i_3j_1j_2j_3} = \mathbf{U}^{(1)}_{i_1j_1}\mathbf{U}^{(2)}_{i_2j_2}\mathbf{U}^{(3)}_{i_3j_3}, \tag{26}$$

with some positive-definite matrices $\mathbf{U}^{(1)}$, $\mathbf{U}^{(2)}$, $\mathbf{U}^{(3)}$. We will, without loss of generality, fix $\mathbf{U}^{(1)}_{11} = \mathbf{U}^{(2)}_{11} = 1$. Then we can easily obtain,

$$\mathbf{U}^{(3)}_{i_3j_3} = \mathcal{T}_{11i_311j_3};\;\; \mathbf{U}^{(2)}_{i_2j_2} = \frac{\mathcal{T}_{1i_211j_21}}{\mathcal{T}_{111111}};\;\; \mathbf{U}^{(1)}_{i_1j_1} = \frac{\mathcal{T}_{i_111j_111}}{\mathcal{T}_{111111}}. \tag{27}$$

Condition in Eq. 26 is equivalent to,

$$\mathcal{T}_{i_1i_2i_3j_1j_2j_3}\;\mathcal{T}^2_{111111} = \mathcal{T}_{11i_311j_3}\mathcal{T}_{1i_211j_21}\mathcal{T}_{i_111j_111}. \tag{28}$$

It is a straightforward substitution to see that Eq. 26 implies Eq. 28. Assume that Eq. 28 holds. Define $\mathbf{U}^{(1)}_{i_1j_1}, \mathbf{U}^{(2)}_{i_2j_2}, \mathbf{U}^{(3)}_{i_3j_3}$ by Eq. 27. Then Eq. 28 will read as,

$$\begin{aligned}\mathcal{T}_{i_1i_2i_3j_1j_2j_3} &= \mathcal{T}_{11i_311j_3}\;\frac{\mathcal{T}_{1i_211j_21}}{\mathcal{T}_{111111}}\;\frac{\mathcal{T}_{i_111j_111}}{\mathcal{T}_{111111}}\\ &= \mathbf{U}^{(1)}_{i_1j_1}\;\mathbf{U}^{(2)}_{i_2j_2}\;\mathbf{U}^{(3)}_{i_3j_3},\end{aligned} \tag{29}$$

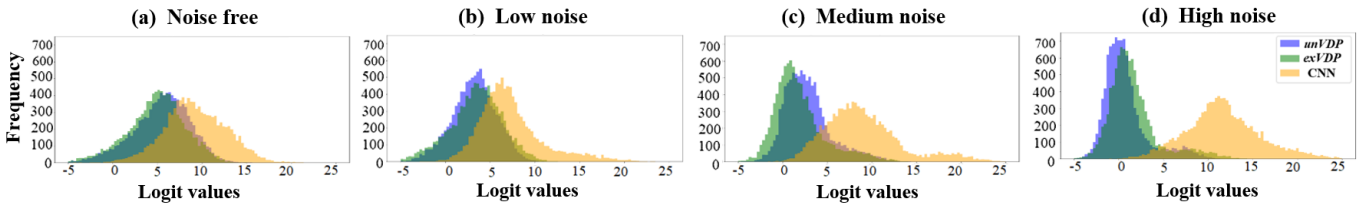

Fig. 9. Histograms of logits at various adversarial noise levels for the unVDP, exVDP and a deterministic CNN trained on the MNIST dataset. The logit values correspond to the correct class. (a) Noise free case. (b) adversarial attack - low noise (c) adversarial attack - medium noise level (d) adversarial attack - high noise level.

Remark 1.

1) *Factor* $\mathfrak{T}^2_{111111}$*, as well as fixation* $\mathbf{U}^{(1)}_{11} = \mathbf{U}^{(2)}_{11} = 1$ *are just normalization parameters.*
2) *Condition in Eq. 28 can be considered in same sense as independence (or rather non-correlation) between three indices of the tensor* $\mathcal{W}_{i_1 i_2 i_3}$*. Indeed, as we notice from Eq. 28, up to normalization factor, the covariance coefficient is the product of covariance coefficients for pairs* $i_1 j_1, i_2 j_2, i_3 j_3$ *with other indices being fixed equal to* 1*.*
3) *The condition in Eq. 28 can be reformulated with respect to any given triple of indices. More precisely, let indices* abc *be fixed, then Eq. 28 is equivalent to Eq. 30. The proof is a straightforward substitution.*

$$\mathfrak{T}_{i_1 i_2 i_3 j_1 j_2 j_3}\ \mathfrak{T}^2_{abcabc} = \mathfrak{T}_{abi_3 abj_3}\mathfrak{T}_{ai_2 caj_2 c}\mathfrak{T}_{i_1 bcj_1 bc}. \quad (30)$$

C. Proof of Proposition 2

Let $\mathbf{b}$ and $\mathbf{w}_i$ be the two independent random vectors defined in Section III-D. For every i^{th} vector, we multiply $\mathbf{b}$ and $\mathbf{w}_i$. The resulting vector $\mathbf{f}$ has the mean and covariance whose entries are derived as follows:

$$\mu_{f_i} = E[\mathbf{w}_i^T\mathbf{b}] = E[\text{tr}(\mathbf{w}_i^T b)] = \text{tr}(E[\mathbf{w}_i^T b]) = \mathbf{m}_i^T\boldsymbol{\mu}_\mathbf{b}, \quad (31)$$

$$\begin{aligned}
\sigma^2_{f_i} &= Var[\mathbf{w}_i^T\mathbf{b}] = E[\text{tr}(\mathbf{w}_i^T\mathbf{b}\mathbf{b}^T\mathbf{w}_i)] - E[\mathbf{w}_i^T\mathbf{b}]E[\mathbf{w}_i^T\mathbf{b}] \\
&= E[\text{tr}(\mathbf{w}_i\mathbf{w}_i^T\mathbf{b}\mathbf{b}^T)] - E[\mathbf{w}_i^T\mathbf{b}]E[\mathbf{w}_i^T\mathbf{b}] \\
&= \text{tr}(E[\mathbf{w}_i\mathbf{w}_i^T]E[\mathbf{b}\mathbf{b}^T]) - E[\mathbf{w}_i^T]E[\mathbf{b}]E[\mathbf{w}_i^T]E[\mathbf{b}] \\
&= \text{tr}\left((\boldsymbol{\Sigma}_i + \mathbf{m}_i\mathbf{m}_i^T)(\boldsymbol{\Sigma}_\mathbf{b} + \boldsymbol{\mu}_\mathbf{b}\boldsymbol{\mu}_\mathbf{b}^T)\right) - \mathbf{m}_i^T\boldsymbol{\mu}_\mathbf{b}\mathbf{m}_i^T\boldsymbol{\mu}_\mathbf{b} \\
&= \text{tr}\left(\boldsymbol{\Sigma}_i\boldsymbol{\Sigma}_\mathbf{b}\right) + \text{tr}(\mathbf{m}_i\mathbf{m}_i^T\boldsymbol{\Sigma}_\mathbf{b}) + \text{tr}(\boldsymbol{\Sigma}_i\boldsymbol{\mu}_\mathbf{b}\boldsymbol{\mu}_\mathbf{b}^T) \\
&\quad + \text{tr}(\mathbf{m}_i\mathbf{m}_i^T\boldsymbol{\mu}_\mathbf{b}\boldsymbol{\mu}_\mathbf{b}^T) - \mathbf{m}_i^T\boldsymbol{\mu}_\mathbf{b}\mathbf{m}_i^T\boldsymbol{\mu}_\mathbf{b} \\
&= \text{tr}\left(\boldsymbol{\Sigma}_i\boldsymbol{\Sigma}_\mathbf{b}\right) + \mathbf{m}_i^T\boldsymbol{\Sigma}_\mathbf{b}\mathbf{m}_i + \boldsymbol{\mu}_\mathbf{b}^T\boldsymbol{\Sigma}_i\boldsymbol{\mu}_\mathbf{b},
\end{aligned} \quad (32)$$

$$\begin{aligned}
\sigma_{f_i f_j} &= Cov[\mathbf{w}_i^T\mathbf{b}, \mathbf{w}_j^T\mathbf{b}] \\
&= E[\text{tr}(\mathbf{w}_i^T\mathbf{b}\mathbf{b}^T\mathbf{w}_j)] - E[\mathbf{w}_i^T\mathbf{b}]E[\mathbf{w}_j^T\mathbf{b}] \\
&= E[\text{tr}(\mathbf{w}_i\mathbf{w}_j^T\mathbf{b}\mathbf{b}^T)] - E[\mathbf{w}_i^T\mathbf{b}]E[\mathbf{w}_j^T\mathbf{b}] \\
&= \text{tr}(E[\mathbf{w}_i\mathbf{w}_j^T]E[\mathbf{b}\mathbf{b}^T]) - E[\mathbf{w}_i^T]E[\mathbf{b}]E[\mathbf{w}_j^T]E[\mathbf{b}] \\
&= \text{tr}\left((\boldsymbol{\Sigma}_{ij} + \mathbf{m}_i\mathbf{m}_j^T)(\boldsymbol{\Sigma}_\mathbf{b} + \boldsymbol{\mu}_\mathbf{b}\boldsymbol{\mu}_\mathbf{b}^T)\right) - \mathbf{m}_i^T\boldsymbol{\mu}_\mathbf{b}\mathbf{m}_j^T\boldsymbol{\mu}_\mathbf{b} \\
&= \text{tr}\left(\boldsymbol{\Sigma}_{ij}\boldsymbol{\Sigma}_\mathbf{b}\right) + \text{tr}(\mathbf{m}_i\mathbf{m}_j^T\boldsymbol{\Sigma}_\mathbf{b}) + \text{tr}(\boldsymbol{\Sigma}_{ij}\boldsymbol{\mu}_\mathbf{b}\boldsymbol{\mu}_\mathbf{b}^T) \\
&\quad + \text{tr}(\mathbf{m}_i\mathbf{m}_j^T\boldsymbol{\mu}_\mathbf{b}\boldsymbol{\mu}_\mathbf{b}^T) - \mathbf{m}_i^T\boldsymbol{\mu}_\mathbf{b}\mathbf{m}_j^T\boldsymbol{\mu}_\mathbf{b} \\
&= \text{tr}\left(\boldsymbol{\Sigma}_{ij}\boldsymbol{\Sigma}_\mathbf{b}\right) + \mathbf{m}_i^T\boldsymbol{\Sigma}_\mathbf{b}\mathbf{m}_j + \boldsymbol{\mu}_\mathbf{b}^T\boldsymbol{\Sigma}_{ij}\boldsymbol{\mu}_\mathbf{b},
\end{aligned} \quad (33)$$

where $i \neq j$. Since we assume that weight vectors $\mathbf{w}_i$, $i = 1, \cdots, H$, in the fully-connected layer are independent, then the cross-covariance matrix $\boldsymbol{\Sigma}_{ij}$ between any pairwise vectors $\mathbf{w}_i$ and $\mathbf{w}_j$, where $i, j = 1, \cdots, H$, is the zero matrix. Then, the covariance between pairwise elements of the vector $\mathbf{f}$ turns out to be,

$$\sigma_{f_i f_j} = \mathbf{m}_i^T\boldsymbol{\Sigma}_\mathbf{b}\mathbf{m}_j, \ i \neq j. \quad (34)$$

D. Covariance Analysis on MNIST and CIFAR-10 Datasets

In Figs. 10 (a and b), we compute the average predictive variance over all test examples at various Gaussian and adversarial noise levels for the CIFAR-10 dataset. The noise level is plotted as the signal-to-noise ratio (SNR) for both Gaussian and adversarial noise in Fig. 10. We also show the test accuracy corresponding to each noise level for both the exVDP and unVDP models in Fig. 10 (c and d). We notice that the average predictive variance increases with decreasing SNR ($\leq$ 25 dB), and the rate of increase in the predictive variance is faster for the adversarial noise as compared to Gaussian noise (from right to left in Figs. 10 (a and b)). The proposed models are self-aware of the noisy conditions, especially when an adversarial attack is expected.

In Fig. 11, we plot the average predictive variance of both the exVDP and unVDP versus SNR for higher levels of Gaussian noise (SNR from 35 dB to -1 dB) than the levels used in Fig. 7 on the MNIST dataset. We note that for SNR < 5, the slope of increase in the predictive variance is steeper, and for SNR ≤ 0, the predictive variance is ≥ 7 for both the exVDP and unVDP models.

ACKNOWLEDGMENT

This work was supported by the National Science Foundation Awards ECCS-1903466, CCF-1527822, and OAC-2008690. Dimah Dera was also supported by an ACM SIGHPC/Intel Computational and Data Science Fellowship Award. 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.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of the 25th International Conference on Neural Information Processing Systems, (NIPS)*, 2012, pp. 1097–1105. 1

[2] B. Omarov and Y. I. Cho, "Machine learning based pattern recognition and classification framework development," in *Proceedings of the 17th International Conference on Control, Automation and Systems (ICCAS)*, 2017, pp. 1–5. 1

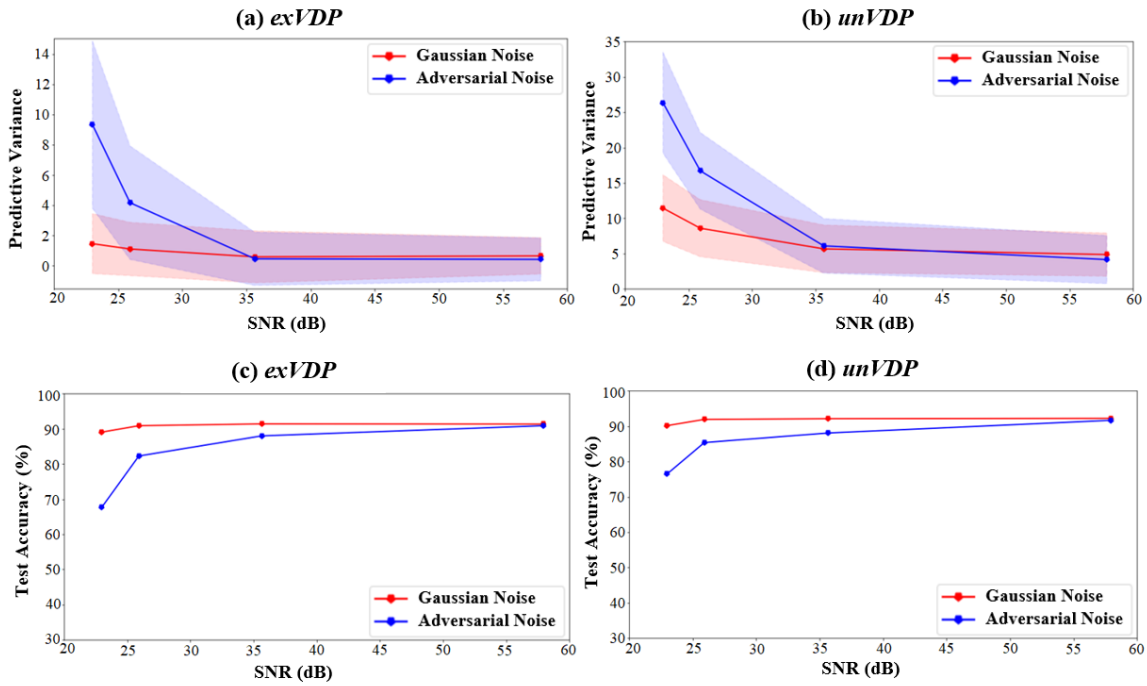

Fig. 10. (a) and (b) Average variance values are plotted against the signal-to-noise ratio (SNR) for varying levels of Gaussian and adversarial noise for exVDP and unVDP, respectively. The variance values are averaged over all test examples of the CIFAR-10 dataset and the lightly filled areas represent the standard deviation. (c) and (d) Average test accuracy correspond to different noise levels is presented for exVDP and unVDP, respectively.

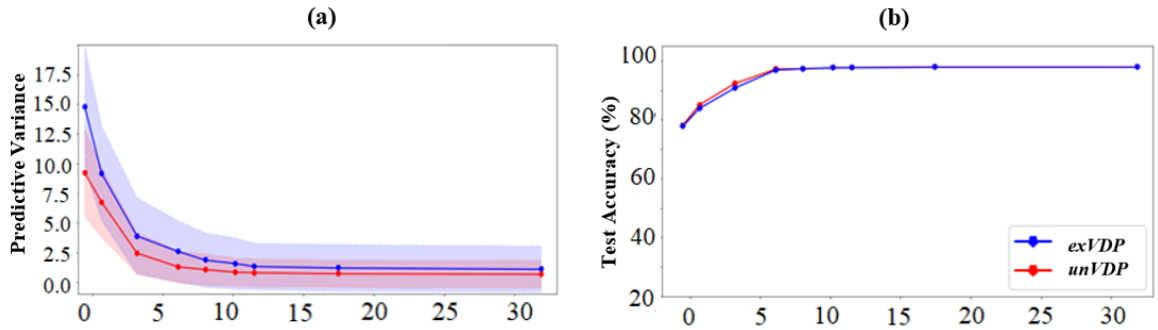

Fig. 11. (a) Average variance values are plotted against the signal-to-noise ratio (SNR) for varying levels of Gaussian noise for the exVDP and unVDP models (blue and red curves, respectively). The variance values are averaged over all test examples of the MNIST dataset and the lightly filled areas represent the standard deviation. (b) Average test accuracy correspond to different noise levels is presented for exVDP and unVDP, respectively.

[3] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proceedings of the 34th International Conference on Machine Learning, (ICML)*, 2017, pp. 1321–1330. 1

[4] D. Yu, J. Li, and l. Deng, "Calibration of confidence measures in speech recognition," *IEEE Transactions on Audio, Speech and Language Processing - TASLP*, vol. 19, pp. 2461–2473, 11 2011.

[5] M. P. Naeini, G. F. Cooper, and M. Hauskrecht, "Obtaining well calibrated probabilities using Bayesian binning," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*. AAAI Press, 2015, p. 2901–2907. 1

[6] G. Scalia, C. A. Grambow, B. Pernici, Y.-P. Li, and W. H. Green, "Evaluating scalable uncertainty estimation methods for deep learning-based molecular property prediction," *Journal of Chemical Information and Modeling*, 2020. 1

[7] F. Kuppers, J. Kronenberger, A. Shantia, and A. Haselhoff, "Multivariate confidence calibration for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 326–327. 1

[8] S. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, 2017, pp. 86–94. 1

[9] A. M. 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, (CVPR)*, 2015, pp. 427–436. 1

[10] J. Ker, L. Wang, J. Rao, and T. Lim, "Deep learning applications in medical image analysis," *IEEE Access*, vol. 6, pp. 9375–9389, 2018. 1

[11] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning, (ICML)*, vol. 37, 2015, pp. 1613–1622. 1, 2, 6

[12] A. Graves, "Practical variational inference for neural networks," in *Proceedings of the 24th International Conference on Neural Information Processing Systems, (NIPS)*, 2011, pp. 2348–2356. 1, 2

[13] J. M. Hernandez-Lobato and R. Adams, "Probabilistic back-propagation for scalable learning of Bayesian neural networks," in *Proceedings of the 32nd International Conference on Machine Learning, (ICML)*, vol. 37, 2015, pp. 1861–1869. 2

[14] S. Ghosh, F. M. D. Fave, and J. S. Yedidia, "Assumed density filtering methods for learning Bayesian neural networks." in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016, pp. 1589–1595. 2

[15] C. Louizos and M. Welling, "Structured and efficient variational deep learning with matrix Gaussian posteriors," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning, (ICML)*, 2016, pp. 1708–1716. 1, 2

[16] H. Ritter, A. Botev, and D. Barber, "A scalable Laplace approximation for neural networks," in *Proceedings of 6th International Conference on Learning Representations, (ICLR)*, 2018. 1, 2

[17] K. Shridhar, F. Laumann, A. Llopart Maurin, and M. Liwicki, "Bayesian convolutional neural networks," *arXiv:1806.05978*, 2018. 1, 2, 6

[18] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J. M. Hernandez-Lobato, and A. L. Gaunt, "Deterministic variational inference for robust Bayesian neural networks," in *Proceedings of 7th International Conference on Learning Representations, (ICLR)*, 2019. 1, 2

[19] Y. Gal and Z. Ghahramani, "Bayesian convolutional neural networks with Bernoulli approximate variational inference," in *Proceedings of 4th International Conference on Learning Representations, (ICLR) workshop track*, 2016. 2, 6

[20] D. Simon, *Optimal State Estimation: Kalman, H Infinity, and Non-linear Approaches*. Wiley-Interscience, 2006. 2, 4, 5

[21] D. Dera, G. Rasool, and N. Bouaynaya, "Extended variational inference for propagating uncertainty in convolutional neural networks," in *IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, Oct 2019, pp. 1–6. 2

[22] S. Julier, "The scaled unscented transformation," in *Proceedings of the 2002 American Control Conference*, vol. 6, 2002, pp. 4555–4559. 2, 5

[23] E. A. Wan and R. Van Der Merwe, "The unscented Kalman filter for non-linear estimation," in *Proceedings of the IEEE Adaptive Systems for Signal Processing, Communications, and Control Symposium*, 2000, pp. 153–158. 2, 5

[24] D. J. C. MacKay, "A practical Bayesian framework for back-propagation networks," *Neural Computation*, vol. 4, no. 3, pp. 448–472, May 1992. 2

[25] R. M. Neal, "Bayesian learning for neural networks," Ph.D. dissertation, University of Toronto, 1995. 2

[26] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Proceedings of the 28th International Conference on International Conference on Machine Learning, (ICML)*, 2011, pp. 681–688. 2

[27] T. Chen, E. Fox, and C. Guestrin, "Stochastic gradient Hamiltonian Monte Carlo," in *Proceedings of the 31st International Conference on Machine Learning, (ICML)*, vol. 32, 2014, pp. 1683–1691. 2

[28] T. Nagapetyan, A. B. Duncan, L. Hasenclever, S. J. Vollmer, L. Szpruch, and K. Zygalakis, "The true cost of stochastic gradient langevin dynamics," *arXiv:1706.02692*, 2017. 2

[29] Y. Li, J. M. Hernández-Lobato, and R. E. Turner, "Stochastic expectation propagation," in *Proceedings of the 28th International Conference on Neural Information Processing Systems, (NIPS)*, 2015, pp. 2323–2331. 2

[30] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the 33th International Conference on International Conference on Machine Learning, (ICML)*, 2016, pp. 1050–1059. 2

[31] W. Roth and F. Pernkopf, "Variational inference in neural networks using an approximate closed-form objective," in *Neural Information Processing Systems, (NIPS) workshop*, 2016. 2

[32] A. H. Abdelaziz, S. Watanabe, J. R. Hershey, E. Vincent, and D. Kolossa, "Uncertainty propagation through deep neural networks," in *Proceedings of the Annual Conference of the International Speech Communication Association, (Interspeech)*, 2015. 2

[33] J. S. Titensky, H. Jananthan, and J. Kepner, "Uncertainty propagation in deep neural networks using extended Kalman filtering," *arXiv:1809.06009*, 2018. 2

[34] W. Yang, D. Dai, and H. Yan, "Feature extraction and uncorrelated discriminant analysis for high-dimensional data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 5, pp. 601–614, 2008. 3

[35] P. Molchanov, S. Tyree, T. Karras, T. Aila, and J. Kautz, "Pruning convolutional neural networks for resource efficient inference," in *Proceedings of 5th International Conference on Learning Representations, (ICLR)*, 2017. 3

[36] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017. 3

[37] A. M. Manceur and P. Dutilleul, "Maximum likelihood estimation for the tensor Normal distribution: Algorithm, minimum sample size, and empirical bias and dispersion," *Journal of Computational and Applied Mathematics*, vol. 239, p. 37 – 49, 2013. 3

[38] A. Papoulis and S. U. Pillai, *Probability, Random Variables, and Stochastic Processes*, 4th ed. McGraw-Hill Higher Education, 2002. 3, 4

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on International Conference on Machine Learning (ICML)*, vol. 37, 2015, p. 448–456. 5

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, 2016, pp. 770–778. 5

[41] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*. The MIT Press, 2005. 5

[42] S. J. Julier and J. K. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Proceedings of SPIE - The International Society for Optical Engineering*, vol. 3068, February 1999. 5

[43] L. Deng, "The MNIST database of handwritten digit images for machine learning research," *IEEE Signal Processing Magazine*, vol. 29, pp. 141–142, 2012. 6

[44] A. Krizhevsky, "Learning multiple layers of features from tiny images," *Master's thesis, Department of Computer Science, University of Toronto*, 2009. 6

[45] Earth Online. PolSAR (The Polarimetric SAR Data Processing and Educational Tool). [Online]. Available: https://earth.esa.int/web/polsarpro/airborne-data-sources 7, 8

[46] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, Y. Burren, N. Porz, J. Slotboom, R. Wiest *et al.*, "The multimodal brain tumor image segmentation benchmark (BRATS)," *IEEE Transactions on Medical Imaging*, vol. 34, no. 10, pp. 1993–2024, 2014. 6, 9

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of 3th International Conference on Learning Representations, (ICLR)*, 2015. 6

[48] 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, (ICLR)*, 2017.

[49] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proceedings of 6th International Conference on Learning Representations, (ICLR)*, 2018. 6

[50] J. M. Duncan, "Factors of safety and reliability in geotechnical engineering," *Journal of Geotechnical and Geoenvironmental Engineering*, vol. 126, no. 4, pp. 307–316, 2000. 6

[51] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," in *Proceedings of 4th International Conference on Learning Representations, (ICLR)*, 2016. 7

[52] Y. Zhou, H. Wang, F. Xu, and Y.-Q. Jin, "Polarimetric SAR image classification using deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 13, no. 12, pp. 1935–1939, 2016. 8

[53] D. Dera, G. Rasool, N. C. Bouaynaya, A. Eichen, S. Shanko, J. Cammerata, and S. 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. 8

[54] N. J. Tustison, B. B. Avants, P. A. Cook, Y. Zheng, A. Egan, P. A. Yushkevich, and J. C. Gee, "N4ITK: improved N3 bias correction," *IEEE Transactions on Medical Imaging*, vol. 29, no. 6, pp. 1310–1320, 2010. 9

[55] S. Pereira, A. Pinto, V. Alves, and C. A. Silva, "Brain tumor segmentation using convolutional neural networks in MRI images," *IEEE Transactions on Medical Imaging*, vol. 35, no. 5, pp. 1240–1251, 2016. 9, 10

[56] H. Kannan, A. Kurakin, and I. Goodfellow, "Adversarial logit pairing," *arXiv:1803.06373*, 2018. 12

[57] A. Shafahi, A. Ghiasi, F. Huang, and T. Goldstein, "Label smoothing and logit squeezing: A replacement for adversarial training?" *arXiv:1910.11585*, 2019. 12

[58] L. Engstrom, A. Ilyas, and A. Athalye, "Evaluating and understanding the robustness of adversarial logit pairing," *arXiv:1807.10272*, 2018. 12

[59] H. Lee, H. Bae, and S. Yoon, "Gradient masking of label smoothing in adversarial robustness," *IEEE Access*, 2020. 12, 13