SDE-Net: Equipping Deep Neural Networks with Uncertainty Estimates

Lingkai Kong [1]   **Jimeng Sun** [2]   **Chao Zhang** [1]

Abstract

Uncertainty quantification is a fundamental yet unsolved problem for deep learning. The Bayesian framework provides a principled way of uncertainty estimation but is often not scalable to modern deep neural nets (DNNs) that have a large number of parameters. Non-Bayesian methods are simple to implement but often conflate different sources of uncertainties and require huge computing resources. We propose a new method for quantifying uncertainties of DNNs from a dynamical system perspective. The core of our method is to view DNN transformations as state evolution of a stochastic dynamical system and introduce a Brownian motion term for capturing epistemic uncertainty. Based on this perspective, we propose a neural stochastic differential equation model (SDE-Net) which consists of (1) a drift net that controls the system to fit the predictive function; and (2) a diffusion net that captures epistemic uncertainty. We theoretically analyze the existence and uniqueness of the solution to SDE-Net. Our experiments demonstrate that the SDE-Net model can outperform existing uncertainty estimation methods across a series of tasks where uncertainty plays a fundamental role.

1. Introduction

Deep Neural Nets (DNNs) have achieved enormous success in a wide spectrum of tasks, such as image classification (Krizhevsky et al., 2012), machine translation (Choukroun et al., 2016), and reinforcement learning (Li, 2017). Despite their remarkable predictive performance, DNNs are poor at quantifying uncertainties for their predictions. Recent studies have shown that DNNs are often overconfident in their predictions and produce mis-calibrated output proba-

[1]School of Computational Science and Engineering, Georgia Institute of Technology, Atlanta, GA [2]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL. Correspondence to: Chao Zhang <chaozhang@gatech.edu>.
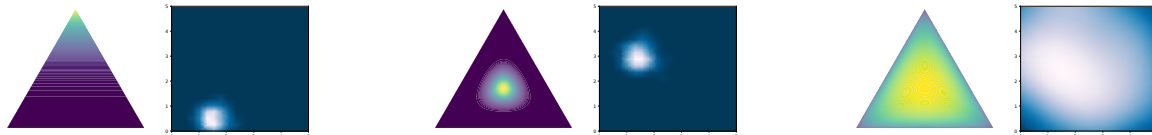
bilities for classification (Guo et al., 2017). Moreover, they can make erroneous yet wildly confident predictions for out-of-distribution samples that are very different from training data (Nguyen et al., 2015). Uncertainty quantification, a key component to equip DNNs with the ability of knowing what they do not know, has become an urgent need for many real-life applications, ranging from self-driving cars to cyber security to automatic medical diagnosis.

Existing approaches to uncertainty quantification for neural nets can be categorized into two lines. The first line is based on Bayesian neural nets (BNNs) (Denker & Lecun, 1991; MacKay, 1992). BNNs quantify predictive uncertainty by imposing probability distributions over model parameters instead of using point estimates. While BNNs provide a principled way of uncertainty quantification, exact inference of parameter posteriors is often intractable. Moreover, specifying parameter priors for BNNs is challenging because the parameters of DNNs are huge in size and uninterpretable.

Along another line, several non-Bayesian approaches have been proposed for uncertainty quantification. The most prominent idea in this line is model ensembling (Lakshminarayanan et al., 2017), which trains multiple DNNs with different initializations and uses their predictions for uncertainty estimation. However, training an ensemble of DNNs can be prohibitively expensive in practice. Other non-Bayesian methods (Geifman et al., 2019) suffer from the drawback of conflating *aleatoric uncertainty*—the natural randomness inherent in the task, with *epistemic uncertainty*—the model uncertainty caused by lack of observation data. In many tasks, it is important to separate these two sources of uncertainties. Taking active learning as an example, one would prefer to collect data from regions with high epistemic uncertainty but low aleatoric uncertainty (Hafner et al., 2018).

We propose a deep neural net model for uncertainty quantification based on *neural stochastic differential equation*. Our model, named SDE-Net, enjoys a number of benefits compared with existing methods: (1) It explicitly models *aleatoric uncertainty* and *epistemic uncertainty* and is able to separate the two sources of uncertainties in its predictions; (2) It is efficient and straightforward to implement, avoiding the need of specifying model prior distributions and inferring posterior distributions as in BNNs; and (3) It

(a) Low aleatoric uncertainty, low epistemic uncertainty.

(b) High aleatoric uncertainty, low epistemic uncertainty.

(c) High aleatoric uncertainty, high epistemic uncertainty.

Figure 1. Different behaviors of a probabilistic model under aleatoric and epistemic uncertainties for classification and regression tasks. The heat maps represent the distributions of model's predictive distributions. The triangles represent classification simplexes and the squares represent regression parameter spaces (x-axis is the predictive mean $\mu(\boldsymbol{x}^*)$; y-axis is the predictive variance $\sigma(\boldsymbol{x}^*)$).

is applicable to both classification and regression tasks.

Our model design (Section 3) is motivated by the connection between neural nets and dynamical systems. From the dynamical system perspective, the forward passes in DNNs can be viewed as state transformations of a dynamic system, which can be defined by an NN-parameterized ordinary differential equation (ODE) (Chen et al., 2018). However, neural ODE is *deterministic* and cannot capture any uncertainty information. In contrast, our model characterizes the transformation of hidden states with *stochastic differential equation* (SDE) and adds a Brownian motion term to explicitly quantify epistemic uncertainty. Our proposed SDE-Net model thus consists of (1) a drift net that parameterizes a differential equation to fit the predictive function, and (2) a diffusion net that parameterizes the Brownian motion and encourages high diffusion for data outside the training distribution. From a control point of view, the drift net controls the system to achieve good predictive accuracy, while the diffusion net characterizes model uncertainty in a stochastic environment. We theoretically analyze the existence and uniqueness of solution to the proposed stochastic dynamical system, which provides insights to design a more efficient and stable network architecture.

Empirical results are presented in Section 4. We evaluate four tasks where uncertainty plays a fundamental role: out-of-distribution detection, misclassification detection, adversarial samples detection and active learning. We find that SDE-Net can outperform state-of-the-art uncertainty estimation methods or achieve competitive results across these tasks on various datasets.

2. Aleatoric Uncertainty and Epistemic Uncertainty

For supervised learning, we are given a training dataset $\mathcal{D} = \{\boldsymbol{x}_j, y_j\}_{j=1}^N$; we train a model $\mathbf{M}$ parameterized by $\boldsymbol{\theta}$ and use the model $\mathbf{M}$ to make predictions for any new test instance $\boldsymbol{x}^*$. The predictive uncertainty comes from two sources (Kendall & Gal, 2017): *aleatoric uncertainty* and *epistemic uncertainty*. Aleatoric uncertainty represents

the natural randomness (*e.g.*, class overlap, data noise, unknown factors) inherent in the task and cannot be explained away with data; while epistemic uncertainty represents our ignorance about model caused by the lack of observation data and is high in regions lacking training data.

Figure 1 illustrates the behaviors of a probabilistic model under the influence of the two sources of uncertainties: (1) When both aleatoric and epistemic uncertainties are low (Figure 1a), the model outputs confident predictions with low variance. This makes the output distributions sharply concentrate at a simplex corner (for classification) or a small-variance region (for regression); (2) When aleatoric uncertainty is high but epistemic uncertainty is low (Figure 1b), the predictive distributions concentrate around the simplex center or large-variance regions; (3) When epistemic uncertainty is high (Figure 1c), the predictive distributions scatter in a highly diffused way over the classification simplex and the regression parameter space.

Bayesian neural networks (BNNs) model epistemic uncertainty by imposing distributions over model parameters. They are realized by first specifying prior distributions for neural net parameters, then inferring parameter posteriors and further integrating over them to make predictions. Unfortunately, such modeling of epistemic uncertainty has two drawbacks. First, it is difficult to specify the prior distributions since the parameters of DNNs are uninterpretable. Second, exact parameter posterior inference is often intractable due to the large number of parameters in DNNs. Most approaches for learning BNNs fall into one of two categories: variational inference (VI) methods (Blundell et al., 2015; Louizos & Welling, 2017; Wu et al., 2019) and Markov chain Monte Carlo (MCMC) methods (Welling & Teh, 2011; Li et al., 2016). VI methods require one to choose a family of approximating distributions, which may lead to underestimation of true uncertainties. MCMC methods are time-consuming and require maintaining many copies of the model parameters, which can be costly for large NNs. To overcome such drawbacks, we will propose a more direct and efficient way to model uncertainties.

3. Uncertainty Quantification via Neural Stochastic Differential Equation

We propose a new uncertainty aware neural net from the stochastic dynamical system perspective. The proposed method can distinguish the two sources of uncertainties with no need of specifying priors of model parameters and performing complicated Bayesian inference.

3.1. Neural Net as Deterministic Dynamical System

Our approach relies on the connection between neural nets and dynamic systems, which has been investigated in (Chen et al., 2018). As neural nets map an input x to an output y through a sequence of hidden layers, the hidden representations can be viewed as the states of a dynamical system. It is thus possible to define a dynamical system by parameterizing its ordinary differential equation with a neural net. To see this, consider the transformation between layers in ResNet (He et al., 2016):

$$x_{t+1} = x_t + f(x_t, t), \qquad (1)$$

where t is the index of the layer while x_t is the hidden state at layer t. We rearrange this equation as $\frac{x_{t+\Delta t} - x_t}{\Delta t} = f(x_t, t)$ where $\Delta t = 1$. Letting $\Delta t \to 0$, we obtain:

$$\lim_{\Delta \to 0} \frac{x_{t+\Delta t} - x_t}{\Delta t} = \frac{dx_t}{dt} = f(x_t, t) \iff dx_t = f(x_t, t)dt. \qquad (2)$$

The transformations in ResNet can thus be viewed as the discretization of a dynamical system, whose continuous dynamics is given by $f(x_t, t)$. The idea of the neural ODE method (Chen et al., 2018) is to parameterize $f(x_t, t)$ with a neural net and exploit an ODE solver to evaluate the hidden unit state wherever necessary. Such a neural ODE formulation enables evaluating hidden unit dynamics with arbitrary accuracy and enjoys better memory and parameter efficiency.

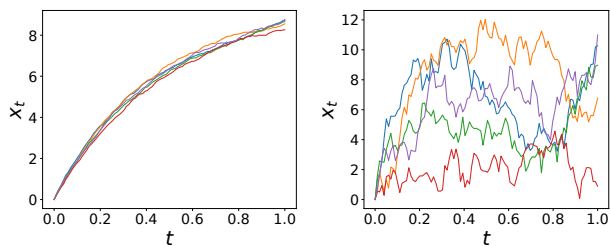## 3.2. Modeling Epistemic Uncertainty with Brownian Motion

However, neural ODE is a deterministic model and cannot model epistemic uncertainty. We develop a neural SDE model to characterize a stochastic dynamical system instead of a deterministic one. The core of our neural SDE model is to capture epistemic uncertainty with *Brownian motion*, which is widely used to model the randomness of moving atoms or molecules in Physics (Bass, 2011).

Definition 3.1 *A standard Brownian motion W_t is a stochastic process which satisfies the following properties: a) $W_0 = 0$; b) $W_t - W_s$ is $\mathcal{N}(0, t - s)$ for all $t \geq s \geq 0$; c) For every pair of disjoint time intervals $[t_1, t_2]$ and $[t_3, t_4]$, with $t_1 < t_2 \leq t_3 \leq t_4$, the increments $W_{t_4} - W_{t_3}$ and $W_{t_2} - W_{t_1}$ are independent random variables.*

We add the Brownian motion term into Eq. (2), which leads to a neural SDE dynamical system. The continuous-time dynamics of the system are then expressed as:

$$dx_t = f(x_t, t)dt + g(x_t, t)dW_t. \qquad (3)$$

Here, $g(x_t, t)$ denotes the variance of the Brownian motion and represents the epistemic uncertainty for the dynamical system. This variance is determined by which region the system is in. As shown in Fig. 2, if the system is in the region with abundant training data and low epistemic uncertainty, the variance of the Brownian motion will be small; if the system is in the region with scarce training data and high epistemic uncertainty, the variance of the Brownian motion will be large. We can thus obtain an epistemic uncertainty estimate from the variance of the final time solution x_T.

(a) System in the region with low uncertainty

(b) System in the region with high uncertainty

Figure 2. 1-D trajectories of a linear SDE for five simulations. When the system is in the region with low uncertainty, *i.e.* small $g(x_t, t)$, the trajectories are more deterministic with small variance. When the system is in the region with high uncertainty, *i.e.* large $g(x_t, t)$, the trajectories are more scattered with large variance.

3.3. SDE-Net for Uncertainty Estimation

As discussed above, we can quantify epistemic uncertainty using Brownian motion. To make the system able to achieve good predictive accuracy and meanwhile provide reliable uncertainty estimates, we design our SDE-Net model to use two separate neural nets to represent the drift and the diffusion of the system as in Fig. 3.

The drift net f in SDE-Net aims to control the system to achieve good predictive accuracy. Another important role of the drift net f is to capture aleatoric uncertainty. This is achieved by representing model output as a probabilistic distribution, *e.g.*, categorical distribution for classification and Gaussian distribution for regression.

The diffusion net g in SDE-Net represents the diffusion of the system. The diffusion of the system should satisfy the following: (1) For regions in the training distribution, the variance of the Brownian motion should be small (low diffusion). The system state is dominated by the drift term in this area and the output variance should be small; (2) For
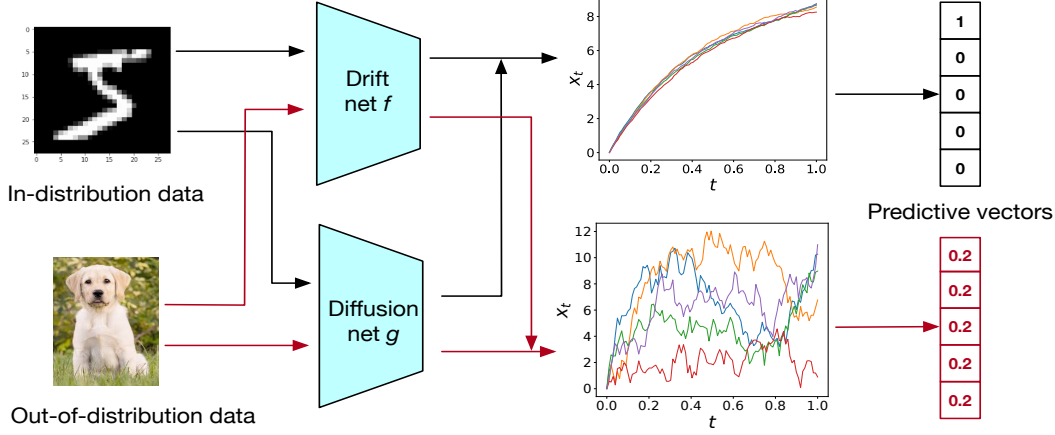
Figure 3. Components of the proposed SDE-Net. For in-distribution data, the system is dominated by the drift net f and achieves good predictive accuracy; for out-of-distribution data, the system is dominated by the diffusion net g and shows high diffusion.

regions outside the training distribution, the variance of the Brownian motion should be large and the system is chaotic (high diffusion). In this case, the variance of the outputs for multiple evaluations should be large.

Based on the above desired properties, we propose the following objective function for training our SDE-Net model:

$$\min_{\boldsymbol{\theta}_f} \mathrm{E}_{\boldsymbol{x}_0 \sim P_{\text{train}}} \mathrm{E}(L(\boldsymbol{x}_T)) + \min_{\boldsymbol{\theta}_g} \mathrm{E}_{\boldsymbol{x}_0 \sim P_{\text{train}}} g(\boldsymbol{x}_0; \boldsymbol{\theta}_g)$$

$$+ \max_{\boldsymbol{\theta}_g} \mathrm{E}_{\tilde{\boldsymbol{x}}_0 \sim P_{\text{OOD}}} g(\tilde{\boldsymbol{x}}_0; \boldsymbol{\theta}_g)$$

$$s.t. \quad d\boldsymbol{x}_t = \underbrace{f(\boldsymbol{x}_t, t; \boldsymbol{\theta}_f)}_{\text{drift neural net}} dt + \underbrace{g(\boldsymbol{x}_0; \boldsymbol{\theta}_g)}_{\text{diffusion neural net}} dW_t, \quad (4)$$

where $L(\cdot)$ is the loss function dependent on the task, *e.g.* cross entropy loss for classification, T is the terminal time of the stochastic process, P_{train} is the distribution for training data, and P_{OOD} is the out-of-distribution (OOD) data. To obtain OOD data, we choose to add additive Gaussian noise to obtain noisy inputs $\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}_0 + \boldsymbol{\epsilon}$ and then distribute the inputs according to the convolved distribution as in (Hafner et al., 2018). An alternative is to use a different, real dataset as a set of samples from the OOD. However, this requires a careful choice of a real dataset to avoid overfitting (Lee et al., 2018).

Unlike the traditional neural nets where each layer has its own parameters, the parameters in our proposed SDE-Net are shared by each layer. This can decrease the number of parameters and leads to significant memory reduction. In the objective function, we also make a simplification that the variance of the diffusion term is only determined by the starting point $\boldsymbol{x}_0$ instead of the instantaneous value $\boldsymbol{x}_t$, which is usually sufficient and can make the optimization procedure easier.

Uncertainty Quantification: Once an SDE-Net is learned,

we can obtain multiple random realizations of the SDE-Net to get samples $\{\boldsymbol{x}_T\}_{m=1}^M$ and then compute the two uncertainties from them. The aleatoric uncertainty is given by the expected predictive entropy $\mathbb{E}_{p(\boldsymbol{x}_T|\boldsymbol{x}_0, \boldsymbol{\theta}_{f,g})}[\mathcal{H}[p(y|\boldsymbol{x}_T)]]$ in classification and expected predictive variance $\mathbb{E}_{p(\boldsymbol{x}_T|\boldsymbol{x}_0, \boldsymbol{\theta}_{f,g})}[\sigma(\boldsymbol{x}_T)]$ in regression. The epistemic uncertainty is given by the variance of the final solution $\mathrm{Var}(\boldsymbol{x}_T)$. This sampling-and-computing operation shares similar spirit with the traditional ensembling method. However, a key difference exists between the two: ensembling methods require training multiple deterministic NNs, while our method just trains one neural SDE model and uses the Brownian motion to encode uncertainty, which incurs much lower time and memory costs.

3.4. Theoretical Analysis

In this subsection, we study the existence and uniqueness of the solution $\boldsymbol{x}_t (0 \leq t \leq T)$ of the proposed stochastic system. Through this theoretical analysis, we can gain insights in designing a more effective network architecture for both the drift net f and the diffusion net g.

Theorem 1 *When there exists $C > 0$ such that*

$$||f(\boldsymbol{x}, t; \boldsymbol{\theta}_f) - f(\boldsymbol{y}, t; \boldsymbol{\theta}_f)|| + ||g(\boldsymbol{x}; \boldsymbol{\theta}_g) - g(\boldsymbol{y}; \boldsymbol{\theta}_g)||$$
$$\leq C||\boldsymbol{x} - \boldsymbol{y}||, \quad \forall \boldsymbol{x}, \boldsymbol{y} \in \mathcal{R}^n, t \geq 0.$$
$$(5)$$

Then, for every $\boldsymbol{x}_0 \in \mathcal{R}^n$, there exists a unique continuous and adapted process $(\boldsymbol{x}_t^{\boldsymbol{x}_0})_{t \geq 0}$ such that for $t \geq 0$

$$\boldsymbol{x}_t^{\boldsymbol{x}_0} = \boldsymbol{x}_0 + \int_0^t f(\boldsymbol{x}_s^{\boldsymbol{x}_0}, t; \boldsymbol{\theta}_f) ds + \int_0^t g(\boldsymbol{x}_0; \boldsymbol{\theta}_g) dW_s \quad (6)$$

Moreover, for every $T \geq 0$, $\mathrm{E}(\sup_{1 \leq s \leq T} |\boldsymbol{x}_s|^2) < +\infty$.

The proof of Theorem 1 can be found in the supplementary material.

Remark. According to Theorem 1, $f(\boldsymbol{x}, t; \boldsymbol{\theta}_f)$ and $g(\boldsymbol{x}; \boldsymbol{\theta}_g)$ must both be uniformly Lipschitz continuous. This can be satisfied by using Lipshitz nonlinear activations in the network architectures, such as ReLU, sigmoid and Tanh (Anil et al., 2019). However, if we naïvely optimize the loss function in Equation (4), $g(\boldsymbol{x}_0; \boldsymbol{\theta}_g)$ can be infinitely large for the input from out-of-distribution. This will lead to explosive solution and make the optimization procedure unstable. To solve this problem, we define the maximum value of the output of $g(\boldsymbol{x}; \boldsymbol{\theta}_g)$ as a hyper-parameter $\sigma_{\max}$. Then, the output of the diffusion neural net is given by a sigmoid function times $\sigma_{\max}$.

3.5. SDE-Net Training

There is no closed form solution to the true final random variable $\boldsymbol{x}_T$. In principle, we can simulate the stochastic dynamics using any high-order numerical solver with adaptive step size (Platen, 1999). However, high-order numerical methods can be costly in the context of deep learning where the input can have thousands of dimensions. Since we focus on supervised learning and uncertainty quantification, we choose to use the simple Euler-Maruyama scheme with fixed step size (Kloeden & Platen, 1992) for efficient network training. Under such a scheme, the time interval $[0, T]$ is divided into N subintervals. Then, we can simulate the SDE by:

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + f(\boldsymbol{x}_k, t; \boldsymbol{\theta}_f)\Delta t + g(\boldsymbol{x}_0; \boldsymbol{\theta}_g)\sqrt{\Delta t}Z_k \quad (7)$$

where $Z_k \sim \mathcal{N}(0, 1)$ is the standard Gaussian random variable and $\Delta t = T/N$. We will show that empirically it suffices to sample only one path for each data point during training time. The number of steps for solving the SDE can be considered equivalently as the number of layers in the definition of traditional neural nets. Then, the training of SDE-Net is actually the forward and backward propagations as in standard neural nets, which can be easily implemented with libraries such as Tensorflow and Pytorch. The drift neural net f and the diffusion neural net g are optimized alternately, as shown in Algorithm 1.

4. Experiments

In this section, we study how the estimated uncertainty can improve model robustness and label efficiency. We first study three tasks on model robustness: (1) out-of-distribution detection, (2) misclassification detection, and (3) adversarial sample detection. We then study how the estimated uncertainties can improve label efficiency on active learning.

Algorithm 1 Training of SDE-Net. h_1 is the downsampling layer; h_2 is the fully connected layer; f and g are the drift net and diffusion net; L is the loss function.

Initialize h_1, f, g and h_2
for # training iterations **do**
    Sample minibatch of N_M data from in-distribution: $\mathbf{X}^{N_M} \sim p_{\text{train}}(x)$
    Forward through the downsampling layer: $\mathbf{X}_0^{N_M} = h_1(\mathbf{X}^{N_M})$
    Forward through the SDE-Net block:
    for $k = 0$ to $N - 1$ **do**
        Sample $\mathbf{Z}_k^{N_M} \sim \mathcal{N}(0, \mathbf{I})$
        $\mathbf{X}_{k+1}^{N_M} = \mathbf{X}_k^{N_M} + f(\mathbf{X}_k^{N_M}, t)\Delta t + g(\mathbf{X}_0^{N_M})\sqrt{\Delta t}\mathbf{Z}_k$
    end for
    Forward through the fully connected layer: $\mathbf{X}_{\text{f}}^{N_M} = h_2(\mathbf{X}_k^{N_M})$
    Update h_1, h_2 and f by $\nabla_{h_1, h_2, f} \frac{1}{N_M} L(\mathbf{X}_{\text{f}}^{N_M})$
    Sample minibatch of N_M data from out-of-distribution: $\mathbf{X}^{N_M} \sim p_{\text{OOD}}(x)$
    Forward through the downsampling layer: $\mathbf{X}_0^{N_M}, \tilde{\mathbf{X}}_0^{N_M} = h_1(\mathbf{X}^{N_M}), h_1(\tilde{\mathbf{X}}^{N_M})$
    Update g by $\nabla_g g(\mathbf{X}_0^{N_M}) - \nabla_g g(\tilde{\mathbf{X}}_0^{N_M})$
end for

4.1. Experimental Setup

We compare our SDE-Net model with the following methods: (1) Threshold (Hendrycks & Gimpel, 2017), which is used in the deterministic DNNs (2) MC-dropout (Gal & Ghahramani, 2016), (3) DeepEnsemble[1] (Lakshminarayanan et al., 2017), (4) Prior network (PN) (Malinin & Gales, 2018), (5) Bayes by Backpropagation (BBP) (Blundell et al., 2015), (6) preconditioned Stochastic gradient Langevin dynamics (p-SGLD) (Li et al., 2016).

The network architecture of the compared methods is a residual net (Chen et al., 2018). For our method, we use one SDE-Net block in place of residual blocks and set the number of subintervals as the number of residual blocks in ResNet for fair comparison—the number of hidden layers in SDE-Net is the same as the baseline models under such settings. For our SDE-Net, we sample one path during training and perform 10 stochastic forward passes at test time in all experiments.

As PN and SDE-Net both involve OOD samples during the training process, we purturb the training data with Gaussian noise (zero mean and variance four for both MNIST and SVHN) as pseudo OOD data. Our supplementary materials provide more details about the implementation, setup, and additional experimental results.

---

[1] We use five neural nets in the ensemble.

Table 1. Classification and out-of-distribution detection results on MNIST and SVHN. All values are in percentage, and larger values indicates better detection performance. We report the average performance and standard deviation for 5 random initializations.

ID	OOD	Model	# Parameters	Classification accuracy	TNR at TPR 95%	AUROC	Detection accuracy	AUPR in	AUPR out
MNIST	SEMEION	Threshold	0.58M	99.5 ± 0.0	94.0 ± 1.4	98.3 ± 0.3	94.8 ± 0.7	99.7 ± 0.1	89.4 ± 1.1
		DeepEnsemble	0.58M × 5	**99.6**± NA	96.0± NA	98.8± NA	95.8± NA	99.8± NA	91.3± NA
		MC-dropout	0.58M	99.5 ± 0.0	92.9 ± 1.6	97.6 ± 0.5	94.2 ± 0.7	99.6 ± 0.1	88.5 ± 1.7
		PN	0.58M	99.3 ± 0.1	93.4 ± 2.2	96.1 ± 1.2	94.5 ± 1.1	98.4 ± 0.7	88.5 ± 1.3
		BBP	1.02M	99.2 ± 0.3	75.0 ± 3.4	94.8 ± 1.2	90.4 ± 2.2	99.2 ± 0.3	76.0 ± 4.2
		p-SGLD	0.58M	99.3 ± 0.2	85.3 ± 2.3	89.1 ± 1.6	90.5 ± 1.3	93.6 ± 1.0	82.8 ± 2.2
		SDE-Net	0.28M	99.4 ± 0.1	**99.6** ± 0.2	**99.9** ± 0.1	**98.6** ± 0.5	**100.0** ± 0	**99.5** ± 0.3
MNIST	SVHN	Threshold	0.58M	99.5 ± 0.0	90.1 ± 2.3	96.8 ± 0.9	92.9 ± 1.1	90.0 ± 3.5	98.7 ± 0.3
		DeepEnsemble	0.58M×5	**99.6**± NA	92.7± NA	98.0± NA	94.1± NA	94.5± NA	99.1± NA
		MC-dropout	0.58M	99.5 ± 0.0	88.7 ± 0.6	95.9 ± 0.4	92.0 ± 0.3	87.6 ± 2.0	98.4 ± 0.1
		PN	0.58 M	99.3 ± 0.1	90.4 ± 2.8	94.1 ± 2.2	93.0 ± 1.4	73.2 ± 7.3	98.0 ± 0.6
		BBP	1.02M	99.2 ± 0.3	80.5 ± 3.2	96.0 ± 1.1	91.9 ± 0.9	92.6 ± 2.4	98.3 ± 0.4
		p-SGLD	0.58M	99.3 ± 0.2	94.5 ± 2.1	95.7 ± 1.3	95.0 ± 1.2	75.6 ± 5.2	98.7 ± 0.2
		SDE-Net	0.28M	99.4 ± 0.1	**97.8** ± 1.1	**99.5** ± 0.2	**97.0** ± 0.2	**98.6** ± 0.6	**99.8** ± 0.1
SVHN	CIFAR10	Threshold	0.58M	95.2 ± 0.1	66.1 ± 1.9	94.4 ± 0.4	89.8 ± 0.5	96.7 ± 0.2	84.6 ± 0.8
		DeepEnsemble	0.58M×5	**95.4**± NA	66.5± NA	94.6± NA	90.1± NA	97.8± NA	84.8± NA
		MC-dropout	0.58M	95.2 ± 0.1	66.9 ± 0.6	94.3 ± 0.1	89.8 ± 0.2	97.6 ± 0.1	84.8 ± 0.2
		PN	0.58M	95.0 ± 0.1	66.9 ± 2.0	89.9 ± 0.6	87.4 ± 0.6	92.5 ± 0.6	82.3 ± 0.9
		BBP	1.02M	93.3 ± 0.6	42.2 ± 1.2	90.4 ± 0.3	83.9 ± 0.4	96.4 ± 0.2	73.9 ± 0.5
		p-SGLD	0.58M	94.1 ± 0.5	63.5 ± 0.9	94.3 ± 0.4	87.8 ± 1.2	97.9 ± 0.2	83.9 ± 0.7
		SDE-Net	0.32 M	94.2 ± 0.2	**87.5** ± 2.8	**97.8** ± 0.4	**92.7** ± 0.7	**99.2** ± 0.2	**93.7** ± 0.9
SVHN	CIFAR100	Threshold	0.58M	95.2 ± 0.1	64.6 ± 1.9	93.8 ± 0.4	88.3 ± 0.4	97.0 ± 0.2	83.7 ± 0.8
		DeepEnsemble	0.58M ×5	**95.4**± NA	64.4± NA	93.9± NA	89.4± NA	97.4± NA	84.8± NA
		MC-dropout	0.58M	95.2 ± 0.1	65.5 ± 1.1	93.7 ± 0.2	89.3 ± 0.3	97.1 ± 0.2	83.9 ± 0.4
		PN	0.58M	95.0 ± 0.1	65.8 ± 1.7	89.1 ± 0.8	86.6 ± 0.7	91.8 ± 0.8	81.6 ± 1.1
		BBP	1.02M	93.3 ± 0.6	42.4 ± 0.3	90.6 ± 0.2	84.3 ± 0.3	96.5 ± 0.1	75.2 ± 0.9
		p-SGLD	0.58M	94.1 ± 0.5	62.0 ± 0.5	91.3 ± 1.2	86.0 ± 0.2	93.1 ± 0.8	81.9 ± 1.3
		SDE-Net	0.32M	94.2 ± 0.2	**83.4** ± 3.6	**97.0** ± 0.4	**91.6** ± 0.7	**98.8** ± 0.1	**92.3** ± 1.1

4.2. Out-of-Distribution Detection

Our first task is out-of-distribution (OOD) detection, which aims to use uncertainty to help the model recognize out-of-distribution samples at test time. In open-world settings, the model needs to deal with continuous data that may come from different data distributions or unseen classes. For OOD samples, it is wiser to let the model say 'I don't know' instead of making an absurdly wrong predictions. We investigate the OOD detection task under both classification and regression settings. Following previous work (Hendrycks & Gimpel, 2017), we use four metrics for the OOD detection task: (1) True negative rate (TNR) at 95% true positive rate (TPR); (2) Area under the receiver operating characteristic curve (AUROC); (3) Area under the precision-recall curve (AUPR); and (4) Detection accuracy. Larger values of them indicate better detection performance.

OOD detection for classification. We first evaluate the performance of different models for OOD detection in classification tasks. For fair comparison, all the methods use the probability of the final predicted class for detection. Table 1 shows the OOD detection performance as well as the classification accuracy on two image classification datasets: MNIST and SVHN. We mix different test OOD datasets with the target dataset (MNIST or SVHN) and evaluate the performance of different models in OOD detection. As shown, SDE-Net consistently achieves the best OOD detection performance among all the models under different combinations. DeepEnsemble is the strongest among the

baselines but it still underperforms SDE-Net consistently. Furthermore, DeepEnsemble needs to train multiple DNNs and incurs much larger computational costs. While PN and SDE-Net both use pseudo OOD data (with Gaussian noise) during training, SDE-Net consistently outperforms PN in all the settings. In addition to using the Gaussian-purturbed OOD data, we also compared the performance of SDE-Net and PN when using real-life OOD datasets during training (see supplementary material). We find that PN is easy to be overfitted, while our SDE-Net is more robust to the choice of OOD data used for training.
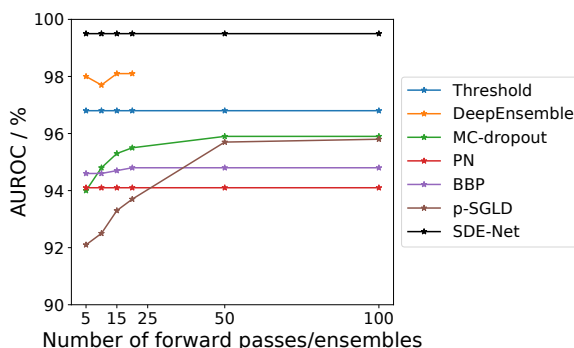

Figure 4. Effect of number of forward passes / ensembles on out-of-distribution (OOD) detection. We use MNIST as the ID data and SVHN as the OOD data.

Fig. 4 shows the impact of the number of forward passes

or ensembles on OOD detection, using MNIST as the ID data and SVHN as the OOD data. As we can see, the BNNs (MC-dropout, p-SGLD and BBP) require more samples than SDE-Net to reach their peak performance at test time. For DeepEnsemble, its performance is already almost saturated when using five nets and larger ensemble sizes can bring little performance gain.

In addition to the OOD detection metrics, we also studied the classification accuracy of different models. We find that the predictive performance of SDE-Net is very close to state-of-the-art results even with significantly fewer parameters. One can further achieve better results by stacking multiple SDE-Net blocks together.

OOD detection for regression. We now investigate OOD detection in regression tasks. Different from classification, few works have studied the OOD detection task for regression. We use the Year Prediction MSD dataset (Dua & Graff, 2017) as training data and the Boston Housing dataset (Bos) as test OOD data. Threshold and PN are excluded here since they only apply to classification tasks. To detect OOD samples for regression tasks, all the methods rely on the variance of the predictive mean. Table 2 shows the OOD detection performance for different methods. The results of other metrics are put into the supplementary material due to the space limit. Because of the imbalance of the test ID and OOD data, AUPR out is a better metric than AUPR in. OOD detection for regression is more difficult than for classification, because regression is a continuous and unbounded problem which makes uncertainty estimation difficult. For this challenging task, all the baselines perform quite poorly, yet SDE-Net still achieves strong performance. The reason is that the diffusion net in SDE-Net directly models the relationship between the input data and epistemic uncertainty, which encourages SDE-Net to output large uncertainty for OOD data and low uncertainty for ID data even for this challenging task.

Table 2. Out-of-distribution detection for regression on Year Prediction MSD + Boston Housing. We report the average performance and standard deviation for 5 random initializations.

Model	# Parameters	RMSE	AUROC	AUPR out
DeepEnsemble	14.9K×5	**8.6**± NA	59.8± NA	1.3± NA
MC-dropout	14.9K	8.7 ± 0.0	53.0 ± 1.2	1.1 ± 0.1
BBP	30.0K	9.5 ± 0.2	56.8 ± 0.9	1.3 ± 0.1
p-SGLD	14.9K	9.3 ± 0.1	52.3 ± 0.7	1.1 ± 0.2
SDE-Net	12.4K	8.7 ± 0.1	**84.4** ± 1.0	**21.3** ± 4.1

4.3. Misclassification Detection

Besides OOD data detection, another important use of uncertainty is to make the model aware when it may make mistakes at test time. Thus, our second task is misclassification detection (Hendrycks & Gimpel, 2017), which aims

Table 3. Misclassification detection performance on MNIST and SVHN. We report the average performance and standard deviation for 5 random initializations.

Data	Model	AUROC	AUPR succ	AUPR err
MNIST	Threshold	94.3 ± 0.9	99.8 ± 0.1	31.9 ± 8.3
	DeepEnsemble	**97.5**± NA	**100.0**± NA	41.4± NA
	MC-dropout	95.8 ± 1.3	99.9 ± 0.0	33.0 ± 6.7
	PN	91.8 ± 0.7	99.8 ± 0.0	33.4 ± 4.6
	BBP	96.5 ± 2.1	**100.0** ± 0.0	35.4 ± 3.2
	P-SGLD	96.4 ± 1.7	**100.0** ± 0.0	**42.0** ± 2.4
	SDE-Net	96.8 ± 0.9	**100.0** ± 0.0	36.6 ± 4.6
SVHN	Threshold	90.1 ± 0.3	99.3 ± 0.0	42.8 ± 0.6
	DeepEnsemble	91.0± NA	**99.4**± NA	46.5± NA
	MC-dropout	90.4 ± 0.6	99.3 ± 0.0	45.0 ± 1.2
	PN	84.0 ± 0.4	98.2 ± 0.2	43.9 ± 1.1
	BBP	91.8 ± 0.2	99.1 ± 0.1	50.7 ± 0.9
	P-SGLD	**93.0** ± 0.4	**99.4** ± 0.1	48.6 ± 1.8
	SDE-Net	92.3 ± 0.5	**99.4** ± 0.0	**53.9** ± 2.5

at leveraging the predictive uncertainty to identify test samples on which the model have misclassified. Table 3 shows the misclassification detection results for different models on MNIST and SVHN. p-SGLD achieves the best overall performance for this task. SDE-Net achieves comparable performance with DeepEnsemble and outperforms other baselines. However, p-SGLD needs to store the copies of the parameters for evaluation, which can be prohibitively costly for large NNs. DeepEnsemble requires training multiple models and incurs high computational cost. Therefore, we argue that SDE-Net is a better choice for the misclassification task in practice.

4.4. Adversarial Sample Detection

Our third task studies adversarial sample detection. Existing works (Szegedy et al., 2014; Goodfellow et al., 2015b) have shown that DNNs are extremely vulnerable to adversarial examples crafted by adding small adversarial perturbations. The ability to detect such adversarial samples is important for AI safety. Different from existing literature on adversarial training, we do not use adversarial training but only examine the uncertainty-aware models' ability in detecting adversarial samples. We study two attacks: Fast Gradient-Sign Method (FGSM) (Goodfellow et al., 2015a), and Projected Gradient Descent (PGD) (Madry et al., 2018).

Fig. 5 shows the detection performance of different models when facing FGSM attacks. As shown, when the perturbation size ϵ varies, SDE-Net can achieve similar AUROC with p-SGLD and outperforms all other methods. On the simpler MNIST dataset, all methods can achieve ∼100% AUROC when the perturbation size is large. However, on the more challenging SVHN dataset, only SDE-Net still converges to 100% AUROC, while other baselines achieve only about 90% AUROC even with perturbation size of one.

Fig. 6 shows the detection performance of different models when facing PGD attacks. We use the default parameters in
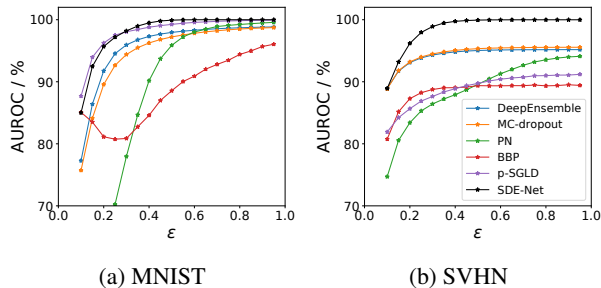
Figure 5. The performance of adversarial sample detection under FGSM attacks. ϵ is the step size in FGSM.
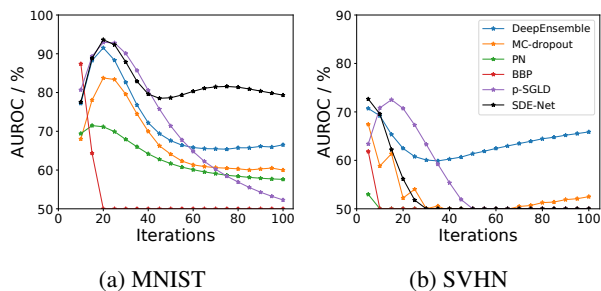

Figure 6. The performance of adversarial sample detection under PGD attacks.

(Madry et al., 2018) and plot the AUROC curve versus the number of PGD iterations. Under the stronger PGD attacks, the AUROCs of all the baselines on MNIST drop below 70% after 60 iterations, while SDE-Net can still achieves over 80% AUROC after 100 iterations. On SVHN, we observe a different picture where all the methods quickly become overconfident except for the costly DeepEnsemble method. This is likely due to higher dimensionality of the data manifold in SVHN. Further work is needed to design efficient and robust uncertainty-aware models that can detect high-dimensional adversarial samples generated by such strong attackers.

4.5. Active Learning

Finally, we study how the estimated uncertainties can improve label efficiency for active learning. Uncertainty plays an important role in active learning. Intuitively, accurate uncertainty estimates can dramatically reduce the amount of labeled data for model training, while inaccurate estimates make the model choose uninformative instances and even lead to worse performance due to overfitting. For active learning, we use the acquisition function proposed in (Hafner et al., 2018):

$$\{\boldsymbol{x}_{\text{new}}, y_{\text{new}}\} \sim p_{\text{new}}(\boldsymbol{x}, y) \propto \left(1 + \frac{\text{Var}[\mu(\boldsymbol{x})]}{\sigma^2(\boldsymbol{x})}\right)^2. \quad (8)$$
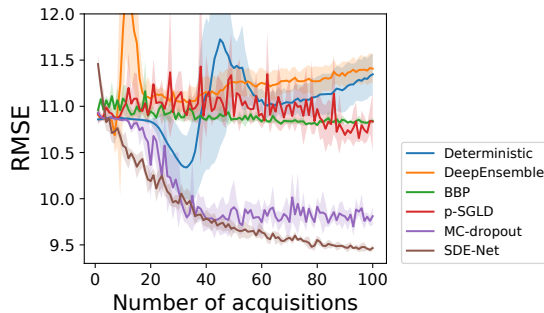

Figure 7. The performance of different models for active learning on the Year Prediction MSD dataset. We report the average performance and standard deviation for 5 random initializations.

This acquisition function allows us to extract the data from the region where the model has high epistemic uncertainty but the data has low aleatoric noise. For the deterministic neural network, we use the predictive variance as a proxy since it cannot model epistemic uncertainty.

We use the Year Prediction MSD regression dataset, where the task is to predict the release year of a song from 90 audio features. It has total 515,345 data points of which 463,715 are for training. We experiment with the following procedure. Starting from 50 labels, the models select a batch of 50 additional labels in every 100 epochs. The remaining data points in the training dataset are available for acquisition, and we evaluate performance on the whole test set.

As we can see from Fig. 7, the RMSE of SDE-Net consistently decreases as we acquire more labeled data. Such results show that SDE-Net successfully acquire data from informative region. However, the performance gain of BBP and p-SGLD are still negligible even after 100 acquisitions. We can also observe that the performance of the deterministic NN and DeepEnsemble start to degrade after several iterations. This is because they keep extracting uninformative data points and thus suffer from overfitting due to the small training data size.

5. Additional Related Work

Uncertainty estimation: BNNs is a principled way for uncertainty quantification. Performing exact Bayesian inference is inefficient and computationally intractable. A common workaround is to use approximation methods like variational inference (Blundell et al., 2015; Louizos & Welling, 2017; Shi et al., 2018; Louizos & Welling, 2016; Zhang et al., 2018), Laplace approximation (Ritter et al., 2018), expectation propagation (Li et al., 2015), stochastic gradient MCMC (Li et al., 2016; Welling & Teh, 2011) and so on. Gal & Ghahramani (2016; 2015) proposed to use

Monte-Carlo Dropout (MC-dropout) at test time to estimate the uncertainty which has a nice interpretation in terms of variational Bayes. Another key element which can affect the performance of BNNs is the choice of prior distribution. The most common prior to use is the independent Gaussian distribution which can only give limited and even biased information for uncertainty. Recently, Hafner et al. (2018) proposed to use noise contrastive priors (NCPs) to obtain reliable uncertainty estimates. Functional variational BNNs (fBNNs) (Sun et al.) employ Gaussian Process (GP) priors and use BNNs for inference.

A number of non-Bayesian methods have also been proposed for uncertainty quantification. DeepEnsemble (Lakshminarayanan et al., 2017) trains an ensemble of NNs and reports competitive uncertainty estimates to MC dropout. Pereyra et al. (2017) adds an entropy penalty as the network regularizer. In (Lee et al., 2018), the authors proposed to minimize a new confidence loss to both a sharp predictive distribution for training data and a flat predictive distribution for OOD data. The OOD data is generated by using a generative model. Prior network (Malinin & Gales, 2018; 2019) parametrized a Dirichlet distribution over categorical output distributions which allows high uncertainty for OOD data, but it is only applicable to classification tasks.

Neural dynamic system: E (2017) first observed the link between ResNet and ODE (Ince, 1956). The residual block which is formulated as $x_{n+1} = x_n + f(x_n)$ can be considered as the forward Euler discretization of the ODE $dx_t = f(x_t)$. In (Lu et al., 2018), the authors show that many state-of-the-art deep network architectures, such as PolyNet (Zhang et al., 2017), FractalNet (Larsson et al., 2017) and RevNet (Gomez et al., 2017), can be regarded as different discretization schemes of ODEs. Chen et al. (2018) further generalized the discrete ResNet to a continuous-depth network by making use of the existing ODE solvers. The adjoint method (Plessix, 2006) is used during ODE-Net training, which allows constant memory cost and adaptive computation. However, these works all focus on improving the predictive accuracy while our work quantifies model uncertainty based on the SDE formulation and the introduced Brownian motion term. Concurrently with this paper, Tzen & Raginsky (2019) establish a connection between infinitely deep residual networks and solutions to SDE. Li et al. (2020) propose a generalization of the adjoint method to compute gradients through solutions of SDEs and apply a latent SDE for continuous time-series data modeling. Our approaches were developed simultaneously but focus on using neural SDEs for uncertainty quantification.

6. Conclusion

We proposed a neural stochastic differential equation model (SDE-Net) for quantifying uncertainties in deep neural nets.

The proposed model can separate different sources of uncertainties compared with existing non-Bayesian methods while being much simpler and more straightforward than Bayesian neural nets. Through comprehensive experiments, we demonstrated that SDE-Net has strong performance compared to state-of-the-art techniques for uncertainty quantification on both classification and regression tasks. To the best of our knowledge, our work represents the first study which establishes the connection between stochastic dynamical system and neural nets for uncertainty quantification. As the approach is general and efficient, we believe this is a promising direction for equipping neural nets with meaningful uncertainties in many safety-critical applications.

Acknowledgements

References

Boston dataset. https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html.

Anil, C., Lucas, J., and Grosse, R. Sorting out lipschitz function approximation. pp. 291–301, 2019.

Bass, R. F. *Stochastic Processes*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2011.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. In *International Conference on Machine Learning*, pp. 1613–1622, 2015.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, pp. 6571–6583, 2018.

Choukroun, S., Cosso, A., et al. Backward sde representation for stochastic control problems with nondominated controlled intensity. *The Annals of Applied Probability*, 26(2):1208–1259, 2016.

Denker, J. and Lecun, Y. Transforming neural-net output levels to probability distributions. In *Advances in Neural Information Processing Systems*, pp. 853–859, 1991.

Dua, D. and Graff, C. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

E, W. A proposal on machine learning via dynamical systems. *Communications in Mathematics and Statistics*, 5: 1–11, 02 2017.

Gal, Y. and Ghahramani, Z. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *International Conference on Machine Learning*, pp. 1050–1059, 2016.

Geifman, Y., Uziel, G., and El-Yaniv, R. Bias-reduced uncertainty estimation for deep neural classifiers. In *International Conference on Learning Representations*, 2019.

Gomez, A. N., Ren, M., Urtasun, R., and Grosse, R. B. The reversible residual network: Backpropagation without storing activations. In *Advances in Neural Information Processing Systems*, pp. 2214–2224, 2017.

Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015a.

Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. 2015b.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *International Conference on Machine Learning*, pp. 1321–1330, 2017.

Hafner, D., Tran, D., Lillicrap, T., Irpan, A., and Davidson, J. Noise contrastive priors for functional uncertainty. *arXiv preprint arXiv:1807.09289*, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. pp. 770–778, 2016.

Hendrycks, D. and Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *International Conference on Learning Representations*, 2017.

Ince, E. *Ordinary Differential Equations*. Courier Corporation, 1956. ISBN 0486603490.

Kendall, A. and Gal, Y. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems*, pp. 5574–5584, 2017.

Kloeden, P. E. and Platen, E. *Numerical solution of stochastic differential equations*. Springer-Verlag Berlin, 1992.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pp. 1097–1105. 2012.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pp. 6402–6413, 2017.

Larsson, G., Maire, M., and Shakhnarovich, G. Fractalnet: Ultra-deep neural networks without residuals. In *International Conference on Learning Representations*, 2017.

Lee, K., Lee, H., Lee, K., and Shin, J. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International Conference on Learning Representations*, 2018.

Li, C., Chen, C., Carlson, D., and Carin, L. Preconditioned stochastic gradient langevin dynamics for deep neural networks. In *AAAI Conference on Artificial Intelligence*, pp. 1788–1794, 2016.

Li, X., Wong, T.-K. L., Chen, R. T., and Duvenaud, D. Scalable gradients for stochastic differential equations. *arXiv preprint arXiv:2001.01328*, 2020.

Li, Y. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274*, 2017.

Li, Y., Hernández-Lobato, J. M., and Turner, R. E. Stochastic expectation propagation. In *Advances in Neural Information Processing Systems*, pp. 2323–2331, 2015.

Louizos, C. and Welling, M. Structured and efficient variational deep learning with matrix gaussian posteriors. In *International Conference on Machine Learning*, pp. 1708–1716, 2016.

Louizos, C. and Welling, M. Multiplicative normalizing flows for variational bayesian neural networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 2218–2227, 2017.

Lu, Y., Zhong, A., Li, Q., and Dong, B. Beyond finite layer neural networks: Bridging deep architectures and numerical differential equations. In *International Conference on Learning Representations*, 2018.

MacKay, D. J. C. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

Malinin, A. and Gales, M. Reverse kl-divergence training of prior networks: Improved uncertainty and adversarial robustness. In *Advances in Neural Information Processing Systems*, pp. 14520–14531, 2019.

Malinin, A. and Gales, M. J. F. Predictive uncertainty estimation via prior networks. In *Advances in Neural Information Processing Systems*, pp. 7047–7058, 2018.

Nguyen, A., Yosinski, J., and Clune, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 427–436, 2015.

Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., and Hinton, G. Regularizing Neural Networks by Penalizing Confident Output Distributions. In *International Conference on Learning Representations*, 2017.

Platen, E. An introduction to numerical methods for stochastic differential equations. *Acta numerica*, 8:197–246, 1999.

Plessix, R.-E. A review of the adjoint state method for computing the gradient of a functional with geophysical applications. *Geophysical Journal International*, 167:495 – 503, 11 2006.

Ritter, H., Botev, A., and Barber, D. A scalable laplace approximation for neural networks. In *International Conference on Learning Representations*, 2018.

Shi, J., Sun, S., and Zhu, J. Kernel implicit variational inference. In *International Conference on Learning Representations*, 2018.

Sun, S., Zhang, G., Shi, J., and Grosse, R. Functional variational bayesian neural networks. In *International Conference on Learning Representations*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

Tzen, B. and Raginsky, M. Neural stochastic differential equations: Deep latent gaussian models in the diffusion limit. *arXiv preprint arXiv:1905.09883*, 2019.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *International Conference on International Conference on Machine Learning*, pp. 681–688, 2011.

Wu, A., Nowozin, S., Meeds, E., Turner, R. E., Hernandez-Lobato, J. M., and Gaunt, A. L. Deterministic variational inference for robust bayesian neural networks. In *International Conference on Learning Representations*, 2019.

Zhang, G., Sun, S., Duvenaud, D., and Grosse, R. Noisy natural gradient as variational inference. In *International Conference on Machine Learning*, pp. 5852–5861, 2018.

Zhang, X., Li, Z., Loy, C. C., and Lin, D. Polynet: A pursuit of structural diversity in very deep networks. *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3900–3908, 2017.