2021 IEEE Winter Conference on Applications of Computer Vision (WACV) | 978-1-6654-0477-8/20/$31.00 ©2021 IEEE | DOI: 10.1109/WACV48630.2021.00410

Improving Robustness and Uncertainty Modelling in Neural Ordinary Differential Equations

Srinivas Anumasa
Computer Science and Engineering
Indian Institute of Technology Hyderbad, India
cs16resch11004@iith.ac.in

P.K. Srijith
Computer Science and Engineering
Indian Institute of Technology Hyderbad, India
srijith@cse.iith.ac.in

Abstract

Deep learning models such as Resnets have resulted in state-of-the-art accuracy in many computer vision problems. Neural ordinary differential equations (NODE) provides a continuous depth generalization of Resnets and overcome drawbacks of Resnet such as model selection and parameter complexity. Though NODE is more robust than Resnet, we find that NODE based architectures are still far away from providing robustness and uncertainty handling required for many computer vision problems. We propose novel NODE models which address these drawbacks. In particular, we propose Gaussian processes (GPs) to model the fully connected neural networks in NODE (NODE-GP) to improve robustness and uncertainty handling capabilities of NODE. The proposed model is flexible to accommodate different NODE architectures, and further improves the model selection capabilities in NODEs. We also find that numerical techniques play an important role in modelling NODE robustness, and propose to use different numerical techniques to improve NODE robustness. We demonstrate the superior robustness and uncertainty handling capabilities of proposed models on adversarial attacks and out-of-distribution experiments for the image classification tasks.

1. Introduction

The ability of deep learning models to capture rich representations of high dimensional data has lead to successful application in computer vision problems like image classification [15, 13], image captioning [25]. The backbone of many of the recent computer vision tasks are deep learning mode1s such as Resnets [13]. They allowed deep learning models to solve complex computer vision tasks by training deep neural networks with more than 100 layers without suffering from vanishing gradient problem. Resnets achieve this using residual connections where input at any layer is added to the output of that layer.

Recently, generalization of Resnet based models was introduced, inspired by ordinary differential equations (ODE) [4, 10]. An ODE parameterized by a neural network can be seen to generalize the Resnets to arbitrarily or infinitely many layers. Such neural ODE (NODE) models have been shown to achieve performance close to Resnet but with a smaller number of parameters. In addition, model selection is also easy with these models as the numerical solver for ODE can automatically determine the number of layers. Like in Resnets, the representations learnt through NODE block are finally mapped to the output through a fully connected neural network (FCNN). NODEs are also shown to be more robust than convolutional neural networks (CNN) [12].

To achieve a generalization performance similar to Resnet architecture, variants of NODE were proposed [10, 5]. They improve generalization performance by concatenating ODE blocks or by adding extra dimensions. When these models are used in high risk domains such as medical [1, 2] and autonomous driving [7], the model should be robust and be able to handle uncertainty well. However, we find that the neural ODE model fails to achieve robustness and uncertainty modelling capabilities required for these real world applications. They perform poorly on adversarial attacks which injects adversarial noise to the data [11, 21]. They may classify an out-of-sample data to a wrong class with high probability. We propose to address these drawbacks in Neural ODE through the use of Bayesian non-parametric approaches like Gaussian processes and through more stable numerical solvers for NODEs.

Bayesian models exhibit good robustness and excellent uncertainty modelling capabilities [9, 8]. In particular, Gaussian processes (GPs) [26] are useful for modelling uncertainty due to their fully Bayesian non-parametric nature. GPs can accurately model predictive uncertainty and provide a more meaningful predictive probabilities. Further, it reduces model selection efforts to a great extent and facilitates hyper-parameter estimation through the marginal likelihood. These properties have lead to their use in various

978-1-6654-0477-8/21/$31.00 ©2021 IEEE
DOI 10.1109/WACV48630.2021.00410

real world problems [22], active learning [17] and global optimization such as Bayesian optimization [23].

We propose a flexible and scalable approach to incorporate uncertainty modeling capabilities in NODE by replacing the final fully connected neural networks with GPs (NODE-GPs). Alternatively, this can be seen as a GP whose kernel takes feature representation provided by NODE as input. Such an approach is shown to have very good robustness and uncertainty modeling capabilities without affecting generalization performance of deep learning models [27, 28]. Moreover, this will also contribute towards reducing model selection efforts by avoiding the requirement to select the FCNN architecture. Thus, the proposed approach further elevates the advantages of NODE for deep learning through better uncertainty modelling and reduced model selection effort. Further, the proposed model offers flexibility to incorporate different kinds of NODE architectures such as ANODE [10] and numerical techniques easily.

To the best of our knowledge, the role of numerical methods on robustness capabilities of NODE have not been studied in the literature before. We investigate different numerical techniques and show that the robustness properties of the NODE are affected by the numerical method used. We demonstrate that NODE using higher order numerical methods are more robust. To trade-off robustness and computational cost, we propose a mixed order numerical technique for NODE with different NODE blocks using different order numerical methods. The proposed mixed order numerical technique for NODE which when combined with GPs showed a superior robustness and uncertainty modeling capabilities on adversarial attack and out-of-distribution (OOD) experiments on image classification problems.

Our contributions can be summarized as follows:

- Propose a novel model which combines NODE based architecture and GP (NODE-GP) to improve uncertainty handling in NODE.
- Demonstrating that numerical method affects NODE robustness and proposing a mixed order numerical method for NODE to improve its robustness.
- Using mixed order numerical technique in NODE-GPs and demonstrating the superior robustness and uncertainty modelling capabilities through adversarial attacks and OOD experiments.

In the following section, we introduce the necessary background required, followed by proposed methodology and experimental setup. In the experiments section, we demonstrate the robustness and uncertainty handling capabilities of the proposed model on image classification.

2. Related Work

NODE [4] is a generalization of Resnet [13] as continuous approximation of intermediate feature vectors of a Residual block. In [4] a memory efficient adjoint method was proposed for computing the gradients of parameters. This approach lead to a series of works [10, 5, 6] by addressing the issues in [4]. Stochastic variants of NODE [16, 20] were also proposed but the evaluation of their proposed methods are restricted to simple architectures. To achieve similar generalization performance compared with Resnet architecture, [10] proposed a modified adjoint approach which address the issue of computing incorrect gradients in [4]. The approach [10] allow concatenation of NODE blocks to achieve a similar accuracy compared to an equivalent Resnet architecture, but is restricted to the usage of discrete numerical methods with fixed step-sizes. It was shown in [12] that NODE [4] is robust to adversarial attacks compared to similar size DNN architecture, but again restricted to simple architecture. Although [10] achieves a good generalization performance, we show in our experiments [10] can be easily fooled with adversarial attacks and lack uncertainty modeling capabilities.

3. Background

Let $\mathcal{D} = \{X, \mathbf{y}\} = \{(\mathbf{x_i}, y_i)\}_{i=1}^{N}$ be set of training data points with $\mathbf{x_i} \in \mathcal{R}^D$ and $y_i \in \{1, \ldots, C\}$. We denote the test data point as $(\mathbf{x}_*, y_*)$. The aim is to learn a function which maps input from the data $\mathbf{x}$ to a class label y so that it will have good generalization performance. Let g be the function learnt using a neural network model, while the function learnt using Gaussian processes to be denoted by f. For a deep learning model such as Resnet composed of multiple blocks, we denote a block i as $g_i(\mathbf{x}, \boldsymbol{\theta}_i)$. The hidden layers in a neural network are denoted as $\mathbf{h}_i$. In this section, we will provide background information required to understand the proposed model.

3.1. Ordinary Differential Equation(ODE)

Ordinary differential equations (ODE) occur in many real world problems like predicting the motion of objects, and change in rate of GDP. For instance, an ODE is written as

$$\frac{d\mathbf{s}_t}{dt} = g(t, \mathbf{s}_t) \tag{1}$$

where $\mathbf{s}_t \in \mathbb{R}^d$, $g : \mathbb{R}^{d+1} \rightarrow \mathbb{R}^d$. We will focus on initial value problems (IVP), i.e. given initial state $\mathbf{s}_0$ we want to compute state value $\mathbf{s}_T$ at time T.

$$\mathbf{s}_T = \mathbf{s}_0 + \int_0^T g(t, \mathbf{s}_t) dt \tag{2}$$

Unfortunately, solutions for most of the ODE cannot be obtained in a closed form. Numerical methods such as the

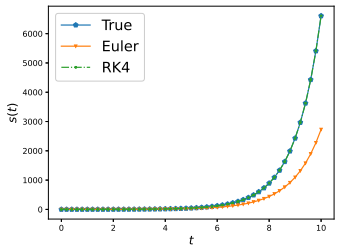


Figure 1. Comparing the solutions computed using Euler and RK4

Euler method and Runge-Kutta4 (RK4) come for the rescue to approximate the solution.

The Euler method is a simple one step method. The parameter required is the step-size dt. The final value $\mathbf{s}_T$ is obtained by iteratively updating the values as

$$\mathbf{s}_{t+1} = \mathbf{s}_t + dt\ g(t, \mathbf{s}_t) \tag{3}$$

RK4 method is a four step method and approximates the solution closer to the actual compared to Euler method.

$$\begin{gathered} \mathbf{k}_1 = g(t, \mathbf{s}_t) \quad ; \quad \mathbf{k}_2 = g(t + \tfrac{dt}{2}, \mathbf{s}_t + dt\tfrac{\mathbf{k}_1}{2}) \\ \mathbf{k}_3 = g(t + \tfrac{dt}{2}\mathbf{s}_t + dt\tfrac{\mathbf{k}_2}{2}); \mathbf{k}_4 = g(t + dt, \mathbf{s}_t + dt\mathbf{k}_3) \\ \mathbf{s}_{t+dt} = \mathbf{s}_t + dt\tfrac{1}{6}(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4) \end{gathered} \tag{4}$$

Here, $\mathbf{k}_1$ is the slope at point $(t, \mathbf{s}_t)$, $\mathbf{k}_2$ is an estimate of the slope at the midpoint $(t + \frac{dt}{2}, \mathbf{s}_t + dt\frac{\mathbf{k}_1}{2})$ obtained using slope $\mathbf{k}_1$ to step half-way through the time step. Similarly, $\mathbf{k}_3$ and $\mathbf{k}_4$ are obtained and the next value s_{t+dt} is obtained by taking a step towards the weighted average of slopes. To illustrate the differences between RK4 and Euler numerical methods, let us consider a simple ODE $\frac{ds}{dt} = \mathbf{s}_t$ with given initial value at $t = 0$ as $\mathbf{s}_0$. True solution of the given ODE is $\mathbf{s}_t = \exp(t)\mathbf{s}_0$. Figure 1 illustrates the solutions computed using RK4 and Euler method, we can observe RK4 method is overlapping with the true solution whereas solution using Euler method is moving further away from true solution over time.

3.2. Neural Ordinary Differential Equations

Deep learning models such as Resnets learn a sequence of transformation by mapping input $\mathbf{x_i}$ to output $\mathbf{y_i}$. In a Resnet block, computation of a hidden layer representation can be expressed using the following transformation.

$$\mathbf{h}_{t+1} = \mathbf{h}_t + g_t(\mathbf{h}_t, \boldsymbol{\theta}_t) \tag{5}$$

where $\mathbf{h}_t$ is a feature vector with $t \in \{0...T\}$ and g is a neural network parameterized by parameters $\boldsymbol{\theta}_t$. If we use the same transformation at every step, Equation 5 can be written as

$$\mathbf{h}_{t+1} = \mathbf{h}_t + g(\mathbf{h}_t, \boldsymbol{\theta}), \tag{6}$$

and this is equivalent to computing the trajectory of the following ODE using Euler method with step size one.

$$\frac{d\mathbf{h}_t}{dt} = g(\mathbf{h}_t, \boldsymbol{\theta}) \tag{7}$$

NODE [4] has taken a step further by modeling continuous transformation of feature vectors in the limiting case of step size tending to zero. Given initial feature vector $\mathbf{h}_0$, the final feature vector $\mathbf{h}_T$ can be computed by solving the ODE $\mathbf{h}'_t = g(\mathbf{h}_t, \boldsymbol{\theta})$ parameterized by $\boldsymbol{\theta}$. On the obtained feature vector $\mathbf{h}_T$ necessary transformations are applied using a fully connected neural network (FCNN), involving multiple linear mapping and activations to predict class probabilities. Using cross-entropy loss function and stochastic mini-batch gradients the parameters of the model are updated using memory efficient adjoint based approach.

Variants of NODE [10, 5] are proposed which can achieve a generalization capability same as Resnets [13]. It consists of multiple NODE blocks, each NODE block B_i given initial feature vector $\mathbf{h}^i_0$, using an assigned numerical method computes intermediate feature vectors $\mathbf{h}^i_t, t \in \{0, 1, ..T\}$ using the function $g_i(\mathbf{h}^i_t, \boldsymbol{\theta}_i)$ parameterized by $\boldsymbol{\theta}_i$. The feature vector $\mathbf{h}^i_T$ obtained from current block B_i then undergo necessary transformation to match the dimension required for processing in block B_{i+1}. The final feature vector from the final Block is transformed through a FCNN to predict class probabilities. Still, NODE and its variants lack the uncertainty modelling and robustness capabilities as observed in our experiments.

3.3. Gaussian process

Gaussian processes (GPs) provide a Bayesian non-parametric approach to learning functions. A GP prior defined over the function can determine various properties of the functions like their smoothness, and stationarity. This can be specified using the mean function $m(\mathbf{x})$ and covariance function or kernel $k(\mathbf{x}, \mathbf{x}')$ associated with a GP. A commonly used kernel is the radial basis function (RBF) kernel $k(\mathbf{x}, \mathbf{x}') = \exp(-\frac{1}{2\ell^2}||\mathbf{x} - \mathbf{x}'||^2)$, where ℓ^2 (length) represents the wiggliness of the function and is learnt from the data. Assuming $f(\mathbf{x})$ is a function sampled from a zero mean GP, we can write $\mathbf{f} = f(X)$ to follow a Gaussian distribution. $\mathbf{f} \sim \mathcal{N}(0, k(X, X))$

In regression, the output y lies around a latent function $f(\mathbf{x})$ sampled from a GP as $y|f(\mathbf{x}) \sim \mathcal{N}(y; f(\mathbf{x}), \sigma^2 I)$. The posterior distribution over the latent function is obtained by combining the prior over $\mathbf{f}$ and the likelihood through Bayes theorem.

$$p(\mathbf{f}|\mathbf{y}) = \frac{\prod_{n=1}^{N} p\left(y_n|f(\mathbf{x}_n)\right) p(\mathbf{f})}{p(\mathbf{y})} \tag{8}$$

The learning in GPs involves estimating hyper-parameters like ℓ^2 from the marginal likelihood, $p(\mathbf{y}) \sim$

$\mathcal{N}(0, k(X,X) + \sigma^2 I)$. This involves a complexity of $O(N^3)$ due to matrix inversion $(K_{X,X} + \sigma^2 I)^{-1}$ over $N \times N$ matrix, where N is the number of training samples. To avoid the cubic complexity sparse approximations of full GP were proposed which reduces the complexity to $O(M^2 N)$, where $M << N$ is the number of inducing inputs [24]. For the classification task, the likelihood of the model $p(\mathbf{y}|f(\mathbf{x}))$ is a softmax function and models the probability of generating the output given the latent function value. However, due to the non-Gaussian nature of the likelihood, approximate inference techniques such as variational inference are employed to get the marginal likelihood and posterior [14, 3].

3.4. Adversarial Attacks and Uncertainty modelling

In order to study the robustness of models we consider two scenarios : Adversarial attacks and uncertainty modelling on out-of-distribution data. Adversarial attacks aim to generate examples which can fool deep learning models to predict a wrong class. We discuss two adversarial attacks namely Fast Gradient Sign Method (FGSM) [11] and Projective Gradient Descent (PGD) [21] which are helpful to determine robustness of the model.

- **Fast Gradient Sign Method(FGSM)** It is a one step white-box adversarial attack, where the adversary has access to the entire model. Given a sample image $\mathbf{x}_*$, gradient of loss with respect to each pixel $\nabla_{\mathbf{x}_*} L(\boldsymbol{\theta}, \mathbf{x}_*, y_*)$ is computed and a proportion of the computed gradient is added to the original image. Thus, $\tilde{\mathbf{x}}_* = \mathbf{x}_* + \epsilon \; sign(\nabla_{\mathbf{x}_*} L(\boldsymbol{\theta}, \mathbf{x}_*, y_*))$. It involves a hyperparameter ϵ which is the strength of the gradient added to the image [1].

- **Projective Gradient Descent(PGD)** Unlike FGSM attack PGD attack is considered to be a powerful white-box attack. PGD is an iterative algorithm, at every step $\mathbf{x}_*^{t+1}$ it computes the gradients of loss with respect to each pixel $\nabla_{\mathbf{x}_*} L(\boldsymbol{\theta}, \mathbf{x}_*, y_*)$, and a proportion of the computed gradient is added to the original image to get an adversarial image. To avoid too much dissimilarity between the actual and perturbed, the perturbed image is projected back into a feasible set around the actual image based on the provided l_p norm. In our experiments we choose l_∞ norm. We have to provide necessary hyperparameters like number of steps n,radius ϵ used for constructing the feasible set $S(\epsilon)$ around the sample $\mathbf{x}_*$ with induced norm which we choose l_∞ [2]. Assuming $\prod$ as projection operator, $\mathbf{x}_*^{t+1} = \prod_{(\mathbf{x}_* + S(\epsilon))} (\mathbf{x}_*^t + \eta \; sign(\nabla_{\mathbf{x}_*} L(\boldsymbol{\theta}, \mathbf{x}_*, y_*)))$.

[1] As our data set are normalized, we test the model performance on ϵ ranging from 0.05 to 0.3.

[2] In our experiments, we choose $n = 10$, ϵ value varies from 0.05 to 0.3 and the step size η is 0.01.

3.4.1 Uncertainty Modelling

DNN models excel in generalization when the testing sample $(\mathbf{x}_*, y_*)$ are from the same as training data distribution. An ideal model should exhibit the necessary feedback when models are fed with test samples from a different distribution. We want our model to exhibit high uncertainty in this case. We discuss two metrics to measure uncertainty, Variation Ratio and Entropy.

- **Variation Ratio(VR):** It is a measure of spread around the mode of class probabilities. Variation ratio can be computed as follows $VR[\mathbf{x}_*] := 1 - \max_{y_*} p(y_*|\mathbf{x}_*)$. If the datapoint is not from the training data distribution or model is uncertain about its true class, we expect to have lower predictive probability, and consequently a larger VR score.

- **Entropy:** It captures the information content and uncertainties in the predictive distribution. It measures the spread of the probability distribution not just over the maximum class but over all the classes. Entropy is computed as $H[\mathbf{x}_*] := -\sum_c p(y_{*c}|\mathbf{x}_*) \log p(y_{*c}|\mathbf{x}_*))$ and is maximum when the predictive distribution is uniformly distributed across the classes and close to zero when it is high for one particular class.

4. Proposed Methodology

In this section we propose a novel model which inherits the properties of NODE based deep learning models and Bayesian non-parametric Gaussian process. This model addresses the drawbacks of NODE and equivalent Resnet in terms of uncertainty handling and robustness. We also discuss the impact of numerical techniques used in NODE models for robustness and propose different NODE architectures based on this observation.

4.1. Combining Neural ODE with GPs

Bayesian learning models compute the predictions by taking expectation over posterior distribution [9]. This allows them to capture the uncertainty in the model prediction and make them more robust against adversarial attacks. In particular, we consider Bayesian non-parametric models such as Gaussian processes, and an approach to combine the predictive modelling capabilities of GPs into Neural ODEs. Combining neural network representations with GPs were shown to have good uncertainty modelling capabilities [28].

The proposed model NODE-GP replaces the fully connected neural network part in the NODE architecture with Gaussian processes. The neural ODE blocks transforms the input to some latent representation. The latent representation is then passed through the GP layer to predict the output

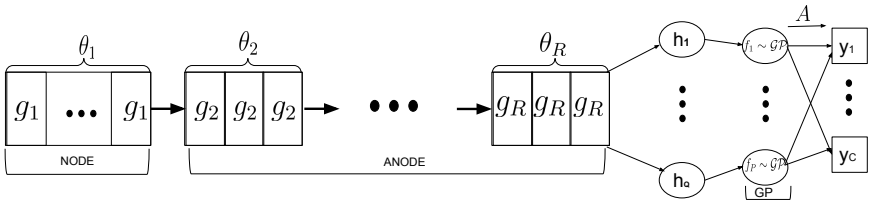


Figure 2. Proposed NODE-GP model combining NODE network with GPs in the final layer.

label. NODE-GPs are quite flexible and can consider any variants of the Neural ODE model before passing it through the final GP layer. Due to the GP final layer, training in NODE-GP is not straightforward. It involves considerable changes in the objective function especially due to the intractable inference in GPs when used for image classification. The loss function considers the GP marginal likelihood or an approximation to it obtained using an appropriate inference technique. The Bayesian non-parametric properties of GPs will also improve model selection capability of NODE and reduce the effective number of parameters. Figure 2 provides an outline of our proposed model where NODE blocks of different complexity are concatenated together with GPs in the final layer to form NODE-GPs.

From the point of view of GPs, the functions sampled from GPs are now defined on a feature vector obtained through multiple NODE block transformations. Let us assume there are R NODE blocks, each with a neural network transformation $g_i(\mathbf{h}_t^i, \boldsymbol{\theta}_i)$, finally resulting in a Q dimensional feature vector $\mathbf{h}_T^R$. Let the final representation over all the data points be represented as H_T^R. Typically, the size of the representation is high and Gaussian processes struggle with a high dimensional data. Following [28], P independent GPs are considered with kernels $k^1, k^2..k^P$ applied on the subsets of features in the Q dimensional feature vector. These GP outputs are then linearly mixed using a matrix A of size $P \times C$ and this captures the correlations across latent function values associated with class labels. The class probabilities are obtained by applying a Softmax function as the likelihood. For a data point, considering one hot encoding of class label as $\mathbf{y}_i \in \{0,1\}^C$, function values from P GPs as $\mathbf{f_i}$, the likelihood is given by

$$p(\mathbf{y}_i|\mathbf{f_i}) = \frac{\mathbf{exp}((\mathbf{Af_i})^\top \mathbf{y_i})}{\sum_{\mathbf{c}} \mathbf{exp}((\mathbf{Af_i})^\top \mathbf{e_c}} \tag{9}$$

where $\mathbf{e_c}$ is an indicator vector with a value of 1 in dimension c and 0 elsewhere.

We use sparse GPs to avoid the $O(N^3)$ complexity of the marginal likelihood. It assumes j^{th} GP with function values over data points $\mathbf{f^j}$ ($\mathbf{f}^j = \{f_{ij}\}_{i=1}^n$) are associated with m inducing points $\mathbf{u}^j$ and inducing inputs Z. The prior can be represented using inducing points as

$$p(\mathbf{f}^j|\mathbf{u}^j) = \mathcal{N}(\mathbf{f}^j|k^j(H_T^R, Z)k^j(Z,Z)^{-1}\mathbf{u}^j, \tilde{K}^j), \tag{10}$$
$$\tilde{K}^j = k^j(H_T^R, H_T^R) - k^j(H_T^R, Z)k^j(Z,Z)^{-1}k^j(Z, H_T^R)$$

As the likelihood (9) is non-Gaussian the posterior cannot be computed in closed form. To address this, variational inference is considered which also allows stochastic gradient training [28]. It assumes the variational posterior over inducing points are factorized over independent GPs as $q(\mathbf{u}) = \prod_j N(\mathbf{u}^j|\boldsymbol{\mu}^j, \mathbf{S}^j)$, where $\boldsymbol{\mu}^j$ are variational mean and $\mathbf{S}^j$ are covariance matrix parameters to be learnt. The corresponding variational lower bound which forms the objective function for training NODE-GPs can be written as

$$\log p(\mathbf{y}) \geq E_{q(\mathbf{u})p(\mathbf{f}|\mathbf{u})}[\log p(\mathbf{y}|\mathbf{f})] - KL[q(\mathbf{u})||p(\mathbf{u})]. \tag{11}$$

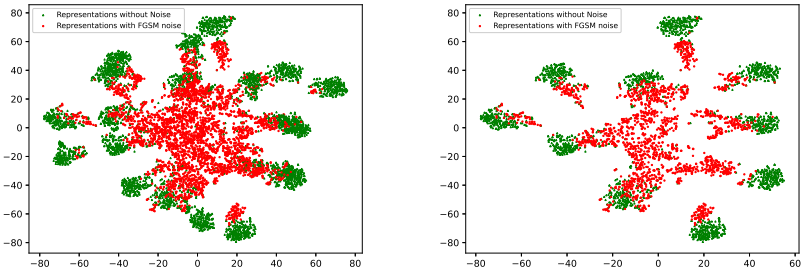
The likelihood can be factorized over data instances as $\log p(\mathbf{y}|\mathbf{f}) = \prod_{i=1}^n [\log p(\mathbf{y}_i|\mathbf{f}_i)]$ which helps in stochastic computation of gradients. The parameters of both NODE model and GP can be learnt jointly by maximizing the lower bound using noisy approximation of gradient on mini batches of data. Equation 11 is intractable due to the expectation term and is approximated using sampling [28]. Predictions are made using a set of inducing point values and consequently latent functions values are sampled from the approximated posterior instead of a point estimate which results in better uncertainty modeling and robustness.

4.2. Numerical Methods and Robustness of NODE

It's found that in ANODE, the choice of numerical method doesn't have much significance in model accuracy [10]. First order methods like Euler or Fourth order like RK4 have the same accuracy, but we observe the models vary in their robustness depending on the numerical method used. In [12], it is shown that NODE [4] is robust towards adversarial attacks compared to CNN of similar architecture. In this NODE model, the intermediate feature vectors are computed using an adaptive numerical method [18] using a higher order numerical method (RK4/RK5) with step size dynamically changing during the forward propagation. In this section, we study and investigate how the order of the numerical technique affects the robustness modelling capabilities in NODE.

Epsilon	0.05	0.1	0.15	0.2	0.25	0.3
NODE_NT2 (RK4)	60.2	54.4	50.05	45.15	40.8	36.2
NODE_NT2 (Euler)	46.8	35.7	31.4	27.7	24.05	19.95

Table 1. Performance of NODE models under FGSM adversarial attack on Cifar10 data. Both the models trained with number of steps 2, Row-1 using Runge-kutta method of order 4 and Row-2 using Euler method



(a) Euler Method (b) RK4 Method

Figure 3. t-sne plots of the features, green points represent feature vectors of clean images and red points represent feature vectors of adversarially affected images (a) ANODE trained with Euler numerical method (b) ANODE trained with RK4 numerical method

To study robustness, in Table 1 we provide performance of NODE models trained with different numerical methods under FGSM adversarial attack [11] on the Cifar10 dataset with increasing *epsilon* values. It can be seen that the performance of ANODE model trained with Euler numerical method degrades more compared to the model trained with RK4 numerical method. To determine the variation in performances of Euler and RK4 based NODEs, we plotted the t-SNE plots of the representations learned by the models in Figure 3. In Figure 3(b), we can see the representations(without noise) learned by the model with RK4 numerical method got clustered into 10 clusters which is equal to the number of classes in the Cifar10 dataset. But, this is not the same for NODE models trained using the Euler method. In Figure 3(a), we do not see a clear 10 clusters as compared to RK4 method. This problem in representation learning without clear distinct boundaries between the classes possibly has led the adversary to easily fool the model. From a numerical method point of view, the solution computed by the Euler method is not close to the actual solution as compared to RK4. Therefore, the path traced by Euler method can be easily altered with a small change in input.

4.2.1 Numerical Methods in NODE-GP

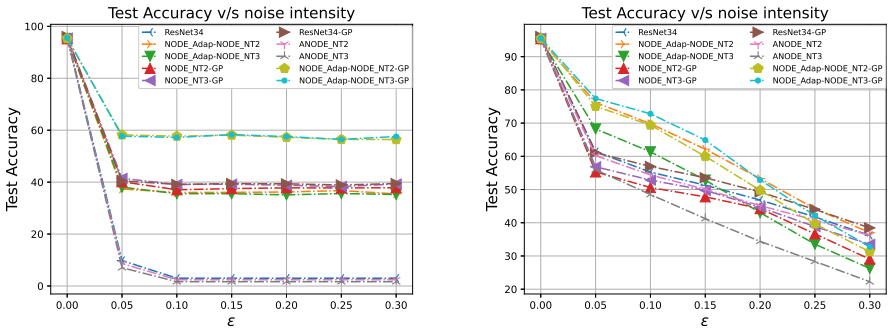
Based on our observation in the previous section, we find that the numerical method used in a NODE plays a role in defining the robustness. But, increasing the order of the numerical method will have an adverse effect on the computational cost. One should trade-off between computational cost and achievable robustness. We consider different variants of the NODE-GP model differing in the numerical method used in the different blocks of the NODE-GP model. Combining the robustness and uncertainty modelling capability of NODE-GP with an appropriate numerical technique will lead to even more robust models as we observe in the experimental section.

5. Experiments

We conducted experiments to test the ability of the proposed models in terms of modelling robustness under adversarial attack and in capturing uncertainty on out-of-distribution samples [3]. We consider different variants of the NODE [4] models and NODE-GP models considering different numerical techniques in different blocks. For all the NODE models time is varied between 0 and 1.

- NODE_NT2 and NODE_NT2-GP: This model uses 4 NODE blocks, with all the NODE blocks using the RK4 numerical solver of two time steps with step size 0.5. The final transformation is done using FCNN in the first and GPs in the second.
- NODE_NT3 and NODE_NT3-GP: Here the number of steps are three instead of two, and step size is 0.33.
- NODE_Adap-NODE_NT2 and NODE_Adap-NODE_NT2-GP: First block uses adaptive numerical method [18] followed by three NODE blocks which uses RK4 numerical method of time steps two, step

[3]Code available at https://github.com/srinivas-quan/NODE-GP.git
[4]NODE block architecture is similar to the one defined in [10]



(a) PGD Attack (b) FGSM Attack

Figure 4. Test accuracy of the models on Cifar10 dataset under (a) PGD attack and (b) FGSM attack

size is 0.5. The two methods differ in the final transformation which is FCNN in the former and GPs in the later.

- NODE_Adap-NODE_NT3 and NODE_Adap-NODE_NT3-GP: Here the number of steps are three instead of two, and step size is 0.33.

Here, NODE_NT2 and NODE_NT3 are the standard NODE model and are considered as the baseline. Our primary objective of experiments is to show improvements in robustness and uncertainty handling capabilities obtained using NODE-GP variants. For completeness, we also compare against standard Resnet34 architecture and its variants using GPs in the final layer (Resnet34-GP) [28].

5.1. Experimental Setup

We conduct our experiments on datasets Cifar10 [19] with 10 classes and Cifar100 [19] with 100 classes. Both the datasets contain 60000 data samples, and we split the samples into 50000 training samples, 5000 validation samples and 5000 samples for testing. Validation dataset is used to choose the model during training.Every model is trained for 350 epochs with varying learning rate similar to the training of Resnet. For the experiments involving adversarial attack, the accuracy mentioned is for 2000 samples from the test data set. For out of distribution experimental setup, the model trained on Cifar10 is tested with samples of 10 classes from Cifar100 such that the class chosen from cifar100 does not match with any of the classes in cifar10.

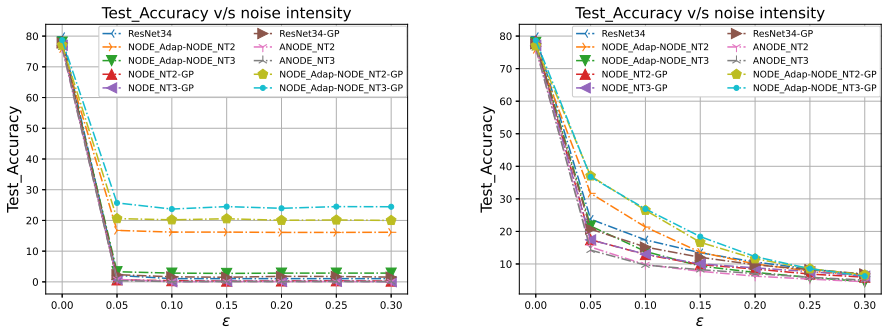
5.2. Adversarial Attacks

We consider two adversarial attack methods FGSM and PGD discussed in Section 3.4 to check the robustness of proposed models. Figure 4(a) and 4(b) shows the plot of test accuracy of the models trained on cifar10 with increasing strength ϵ of the adversarial attacks PGD and FGSM respectively. The results of PGD and FGSM attacks for Cifar100 is shown in Figure 5(a) and 5(b) respectively.

We can observe that in Figure 4 and Figure 5, the models NODE_Adap-NODE_NT2-GP and NODE_Adap-NODE_NT3-GP show better robustness towards attacks compared to all the models. For example, under PGD attack over cifar10 dataset in Figure 4(a), NODE_Adap-NODE_NT3-GP gives 20% more accuracy than NODE_NT3-GP for $\epsilon \geq 0.05$ proving the choice of numerical method matters and gives 22% more accuracy than NODE_Adap-NODE_NT3 for $\epsilon \geq 0.05$ proving that inclusion of GP has helped in improving the robustness. In this case, NODE_Adap-NODE_NT2-GP is 52% more accurate than the baseline standard NODE (NODE_NT2) and NODE_Adap-NODE_NT2 is 30% more accurate than NODE_NT2, demonstrating the superior robustness of the proposed NODE models over the standard NODE model. The predictive distribution in GP considers distribution over function values rather than point estimates, helping in reducing the adversarial noise in the image. The proposed models NODE_Adap-NODE_NT2-GP, NODE_Adap-NODE_NT3-GP perform better than Resnet variants in almost all values of ϵ as observed in Figure 4 and Figure 5. We can observe from Figure 4(a) and Figure 5(a) after an ϵ value we don't see much of a change in accuracy. This is due to the particular nature of the PGD algorithm, where not much change happens to the image after the adversary noise is added and projected back to the feasible set around the image.

5.3. Out-of-Distribution Experiments

We conduct experiments to see the uncertainty modelling capabilities of the proposed models through their performance on out-of-distribution data. For the models trained on cifar10, to test the model uncertainty we feed samples from cifar100 dataset. Figure 6 shows the performance of the models using the metrics Entropy and Variation ratio. For every model, predictive probability is computed on samples chosen from out-of-distribution data and an average Entropy and Variation ratio is computed. Models which



(a) PGD Attack (b) FGSM Attack

Figure 5. Test accuracy of the models on Cifar100 dataset under (a) PGD attack and (b) FGSM attack

Dataset	NODE _NT2	NODE _NT3	Res net34	Res net34 -GP	NODE _Adap- NODE _NT2	NODE _Adap- NODE _NT3	NODE _NT2 -GP	NODE _NT3 -GP	NODE _Adap- NODE _NT2-GP	NODE _Adap- NODE _NT3-GP
cifar10	95.3	95.7	95.3	95.2	95.35	95.45	95.4	95.6	95.4	95.5
cifar100	76.1	76.4	78.3	76.8	76.9	76.5	75.8	77	76.6	76.7

Table 2. Generalization performance of Models in terms of test accuracies on Cifar10 and Cifar100 datasets.

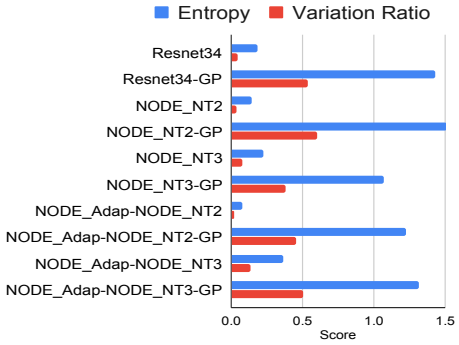


Figure 6. Performance of Models on out-of-distribution setting (trained on Cifar10 and tested on Cifar100) using metrics Entropy and Variation Ratio

use Gaussian processes for final transformation exhibit high Entropy and Variation ratio compared to the models without GP. We find that NODE-GP model, NODE_NT2-GP exhibited the best uncertainty modelling capabilities in terms of uncertainty and variation ratio. In fact, NODE_NT2-GP gave entropy score 1.51 and variation ratio score 0.6 as compared to 0.137 and 0.038 by standard NODE (NODE_NT2). Thus, the proposed NODE-GP models provide much better uncertainty estimates.

As stated before, the main objective of our work is to improve robustness and uncertainty modeling capabilities in NODE. The experimental results showed that we have indeed achieved our objective. But we also study if this will affect the generalization performance of the proposed models. The test accuracy of various models on Cifar10 and Cifar100 data sets are provided in Table 2. We find that the proposed NODE and NODE-GP models achieve an accuracy close to the popular Resnet models. For Cifar10, two of the NODE-GP models gave better performance than standard Resnets. Thus, the proposed models are able to achieve superior robustness and uncertainty modelling capabilities without affecting the generalization performance. Moreover, NODE and NODE-GP models have the advantage of less effort in model selection and smaller number of parameters. NODE-GP involves only 13.2 Million parameters compared to 23.4 Million parameters in Resnet34-GP.

6. Conclusion

Neural ODEs provide a continuous time counterpart to Resnets, with advantages in the form of model selection and reduced number of parameters. However, we found that these models lack robustness and uncertainty handling capabilities required to help decision making in many real world problems. We propose a novel model combining Neural ODEs with GPs which will provide the required robustness and uncertainty handling capabilities. Moreover, the model is flexible enough to accommodate different types of NODEs. We also showed that numerical method plays a role in NODE robustness. We consider various types of NODE architectures and numerical solvers to improve the robustness and uncertainty handling capabilities in NODE-GP. The experimental results on Cifar10 and Cifar100 demonstrated the superior performance of the proposed NODE-GP models against adversarial attacks and uncertainty handling in out-of-distribution data.

References

[1] Saad Ullah Akram, Juho Kannala, Lauri Eklund, and Janne Heikkilä. Cell segmentation proposal network for microscopy image analysis. In *Deep Learning and Data Labeling for Medical Applications*, pages 21–29. Springer, 2016.

[2] Ayelet Akselrod-Ballin, Leonid Karlinsky, Sharon Alpert, Sharbell Hasoul, Rami Ben-Ari, and Ella Barkan. A region based convolutional network for tumor detection and classification in breast mammography. In *Deep learning and data labeling for medical applications*, pages 197–205. Springer, 2016.

[3] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[4] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Advances in neural information processing systems*, pages 6571–6583, 2018.

[5] Emilien Dupont, Arnaud Doucet, and Yee Whye Teh. Augmented neural odes. In *Advances in Neural Information Processing Systems*, pages 3140–3150, 2019.

[6] Conor Durkan, Artur Bekasov, Iain Murray, and George Papamakarios. Neural spline flows. In *Advances in Neural Information Processing Systems*, pages 7511–7522, 2019.

[7] Lex Fridman, Daniel E Brown, Michael Glazer, William Angell, Spencer Dodd, Benedikt Jenik, Jack Terwilliger, Aleksandr Patsekin, Julia Kindelsberger, Li Ding, et al. Mit advanced vehicle technology study: Large-scale naturalistic driving study of driver behavior and interaction with automation. *IEEE Access*, 2019.

[8] Yarin Gal. *Uncertainty in deep learning*. PhD thesis, PhD thesis, University of Cambridge, 2016.

[9] Andrew Gelman, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2nd ed. edition, 2004.

[10] Amir Gholami, Kurt Keutzer, and George Biros. Anode: unconditionally accurate memory-efficient gradients for neural odes. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 730–736. AAAI Press, 2019.

[11] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015*.

[12] YAN Hanshu, DU Jiawei, TAN Vincent, and FENG Jiashi. On robustness of neural ordinary differential equations. In *International Conference on Learning Representations*, 2019.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] James Hensman, Nicolò Fusi, and Neil D. Lawrence. Gaussian processes for big data. In *Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence*, page 282–290, 2013.

[15] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708, 2017.

[16] Junteng Jia and Austin R Benson. Neural jump stochastic differential equations. In *Advances in Neural Information Processing Systems*, pages 9847–9858, 2019.

[17] Ashish Kapoor, Kristen Grauman, Raquel Urtasun, and Trevor Darrell. Gaussian processes for object categorization. *International Journal of Computer Vision*, 88(2):169–188, Jul 2009.

[18] Toshinori Kimura. On dormand-prince method. *Japan Malaysia Technical Institute*, 40(10):1–9, 2009.

[19] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[20] Xuanqing Liu, Tesi Xiao, Si Si, Qin Cao, Sanjiv Kumar, and Cho-Jui Hsieh. How does noise help robustness? explanation and exploration under the neural sde framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 282–290, 2020.

[21] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

[22] Filipe Rodrigues and Francisco C. Pereira. Heteroscedastic gaussian processes for uncertainty modeling in large-scale crowdsourced traffic data. *Transportation Research Part C: Emerging Technologies*, 95:636–651, Oct 2018.

[23] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P. Adams, and Nando de Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, Jan 2016.

[24] Edward Snelson and Zoubin Ghahramani. Sparse gaussian processes using pseudo-inputs. In *Advances in neural information processing systems*, pages 1257–1264, 2006.

[25] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.

[26] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[27] Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P Xing. Deep kernel learning. In *Artificial Intelligence and Statistics*, pages 370–378, 2016.

[28] Andrew G Wilson, Zhiting Hu, Russ R Salakhutdinov, and Eric P Xing. Stochastic variational deep kernel learning. In *Advances in Neural Information Processing Systems*, pages 2586–2594, 2016.