Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа искусственного интеллекта

УДК 007.724.1
Инв. No

Работа допущена к защите

преподаватель

_____Д.Е.Моторин

«_____»_____2023.

Курсовая работа

SDE-Net: Equipping Deep Neural Networks with Uncertainty Estimates

Студент: Эспинола Ривера, Хольгер Элиас
1-й курс магистратура
Искусственый интеллект и машинное обучение

Санкт-Петербург

2023

1. Introduction and Related Works

1.1. Abstract

Purpose and motivation. This research is a purpose of a novel method to estimate an uncertainty in complex deep learning models.

Problem. The traditional methods to estimate the uncertainty in mathematical models like Bayesian and Non-Bayesian approaches don't have the necessary scalability that neural networks require to estimate the uncertainty, having large number of parameters and demanding a lot of computer power. They are not practical methods in this context. The big challenge is find an appropriate and effective method to estimate the uncertainty in deep learning models.

Approach. To estimate uncertainty in deep neural networks, this research takes a dynamical system perspective. The implementation of Stochastic Differential Equation Network (SDE-Net) interpreting Deep Neural Network (DNN) as state evolution of stochastic dynamical system with Brownian motion term.

Results. This research takes experiments in tasks where uncertainty have a fundamental role. The SDE-Net outperforms the capacity to estimate uncertainty in comparison with currently methods.

Conclusion. The SDE-Net experimentally demonstrated to be more effective than traditional approaches to estimate uncertainty.

1.2. Read the introduction + related work

Introduction

Despite the great results achieved by Deep Neural Network models, many times these models make erroneous predictions with high confidence in scenarios where the evidence to support those predictions are not enough (overconfidence). The overconfidence usually produces inaccurate and unreliable results. To avoid this scenario, is necessary quantify the uncertainty and define what these models don't know.

Currently exists 2 approaches to estimate the uncertainty of models: Bayesian and non-Bayesian methods. In the case of Bayesian Methods, to quantify the uncertainty, these models define probability distributions for each parameter of model. Make these estimations and after that, make the inference are not viable way given the huge volume of parameters and the fact to these parameters don't have semantic interpretation. In the case of non-Bayesian methods, in the case of ensemble methods, requires train multiple NN with different initializations. Doing this implies a high computational cost. In other hand, in other non-Bayesian methods presents a drawback of separating aleatoric and epistemic uncertainty.

The purpose of research considering an implementation of Stochastic Differential Equation Net (SDE-Net) to estimate uncertainty in Deep Neural Network (DNN) models.

The motivation to take this approach is sustained in the connection existing between neural networks and dynamical systems. In fact, we can see the DNN like a dynamical system. The forward passes in hidden layers can be interpreted like state of transformations in dynamical system defined by Neural Ordinary Differential Equation (Neural-ODE). Here, we have the deterministic model. To introduce the uncertainty in the model, the introduction of Brownian motion term, capture the epistemic uncertainty of these nets.

The implementation of the SDE-Net has 2 parts:
(1) drift-net that parametrizes a differential equation, taking the deterministic behavior of the model to fit the predictive function.
(2) diffusion-net that parametrizes the Brownian motion term to quantify epistemic uncertainty. Integrating these 2 parts in the network, the SDE-Net will provide the good capacity to do accurate predictions and estimate the uncertainty adding to this model, a deterministic part of model in a stochastic environment.

Related works

a. Blundell, C., Cornebise, J., Kavukcuoglu, K., & Wierstra, D. (2015, June). Weight uncertainty in neural network.

Bayesian approach to estimate uncertainty. This research purpose a backpropagation-algorithm for learning a probability distribution on the weights of a neural network. With this approach and using regularization methods for the weights, this research demonstrates that the learnt of uncertainty in the weights can be used to improve the generalization levels in non-linear regression problems and can be used to drive the exploration-exploitation trade-off in reinforcement learning. The purposed algorithm called Bayes by Backprop optimizes a well-defined objective function to learn a distribution of the weights of a neural network. In the problem of non-linear regression, the study found good and reasonable predictions in unseen data. The results of this regularization are comparable to the Dropout regularization. In the case of the reinforcement learning, was seen automatic learning of trade-off exploration-exploitation.

b. Hafner, D., Tran, D., Lillicrap, T., Irpan, A., & Davidson, J. (2020, August). Noise contrastive priors for functional uncertainty.

Bayesian approach to estimate uncertainty. In this research was proposed Noise Contrastive priors (NCP) to obtain reliable uncertainty estimates. The main idea of this approach is to train the model to take very good predictions in scenarios where have high uncertainty for data points outside of the training data. To do this, NCP are data priors that are enforced on both training inputs x and inputs x' perturbed by a noise. Adding to this, priors in data space can easily capture properties such as periodicity or spatial invariance. NCP can prevent overfitting outside for data outside to training distribution. This approach fixes the currently problem in the models that generally uses independent Gaussian distribution, which give limited and even biased information for uncertainty.

c. Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles.

Non-Bayesian approach to estimate uncertainty. In this research, the study takes in account the problems in the requirements to estimate uncertainty in models like non-flexible training procedure and the huge computational cost, and makes an ensemble of NNs, and with this, reoriented the effort to have a good managing of hyperparameter tuning and with this, obtain a high predictive capacity for uncertainty estimates. The predictive uncertainty was made in test examples to know and unknow distributions in the context of classification and regression tasks. Two sources of uncertainty were captured: in probabilistic NN, notes the ambiguity in outputs (y) given the inputs (x). Adding to this, the combination of ensembles takes the averaging predictions over the multiple models which had different initializations, capturing the model uncertainty. With this work, was seen that non-Bayesian approach to estimate uncertainty can provide good predictive uncertainty estimates for the previously mentioned tasks.

d. Chen, R. T., Rubanova, Y., Bettencourt, J., & Duvenaud, D. K. (2018). Neural ordinary differential equations.

In this research was introduced a deep learning model called Ordinary Differential Equation Network (ODE-Network). This model parametrizes the derivates of the hidden states using a deep neural network and demonstrated that residual neural networks and ODE-Networks present very similar behavior, that suggest the behavior of dynamical systems in these nets. Under this ODE-Network, was implemented new models for time-series modelling, supervised learning and density estimation.

Limitations

All the methods discussed in the related works dealt the problem of the estimation of uncertainty. With the Bayesian approaches, occurs the problem of the huge computation necessary to scale these methods to more complex deep neural nets, which makes it intractable. In other way, the selection of prior gaussian distribution is a problem solved by NCP. In the case of the non-Bayesian approach, makes ensemble of multiple deep neural nets with huge volume of parameters is a limitation for the scalability of these methods for estimate uncertainty in more and more complex models in the currently situation where the trend of the models is to complexity increasing. In the case of the ODE-Network, is interesting the fact to stablish relationship between dynamic systems and deep neural nets, but this model considers just the deterministic part of the problem of the modeling, and this approach needs extension to problem of uncertainty under the models.

2. Model and Method

2.1.List of Annotations

x_t: is the hidden state at layer t
t: index of the hidden layer
$f(x_t, t)$: function of continuous dynamic in the system
dx_t: rate of change in hidden state at layer t
dt: rate of change of indexes of the hidden layer
$g(x_t, t)$: variance of the Brownian motion
W_t: standard Brownian motion
E: mathematic expectation
L(.): loss function – dependently of the task
P_{train}: probability distribution of training data
$P_{OOD:}$ out-of-distribution data
x_0': noisy inputs - inputs added with gaussian noise
Z_k: standard Gaussian random variable ($Z_k \sim \aleph(0,1)$)

2.2. Mathematical Model

The research purpose SDE-Net to estimate uncertainty, taking a stochastic dynamical system perspective and explicitly distinguishing the 2 sources of uncertainty.

The starting point for modeling the problem is take in account the model of transformation between layers in ResNet:

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

Where:

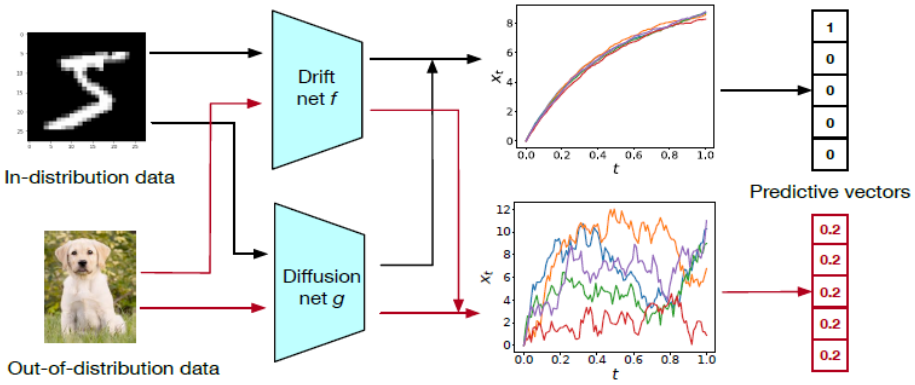
$$\lim_{x \to 0} \frac{x_{t+\Delta t} - x_t}{\Delta t} = \frac{dx_t}{dt} = f(x_t, t) \Leftrightarrow dx_t = f(x_t, t)dt \tag{2}$$

The Eq. (2) is considered the ODE in dynamical systems. However, this deterministic model doesn't model epistemic uncertainty. To capture the epistemic uncertainty, we add the Brownian motion term and the Stochastic Differential Equation is expressed as:

$$dx_t = \underbrace{f(x_t, t)}_{\textit{drift net}} dt + \underbrace{g(x_t, t)}_{\textit{diffusion net}} dW_t \tag{3}$$

With the variance of the Brownian motion, we can control the level of epistemic uncertainty. Scenarios where exists abundant training data and low epistemic uncertainty, the variance of the Brownian motion will be small. In the case when the training data is scarce and have high epistemic uncertainty, the variance of the Brownian motion will be high. This information is contained in $g(x_t, t)$.

2.3. SDE-Net Architecture



The SDE-Net have 2 separated neural networks:

Drift-Net: In Stochastic differential equation represented by Eq. (3), this net model the function *f*. Here, we have the control to system achieve good results. This model captures the aleatoric uncertainty.

Diffusion-Net: In Stochastic differential equation represented by Eq. (3), this net model the function *g*. Here, we have the diffusion of the system. Maybe deterministic and dominated by the drift for small variance of Brownian motion or chaotic and dominated by the diffusion for high variance of Brownian motion.

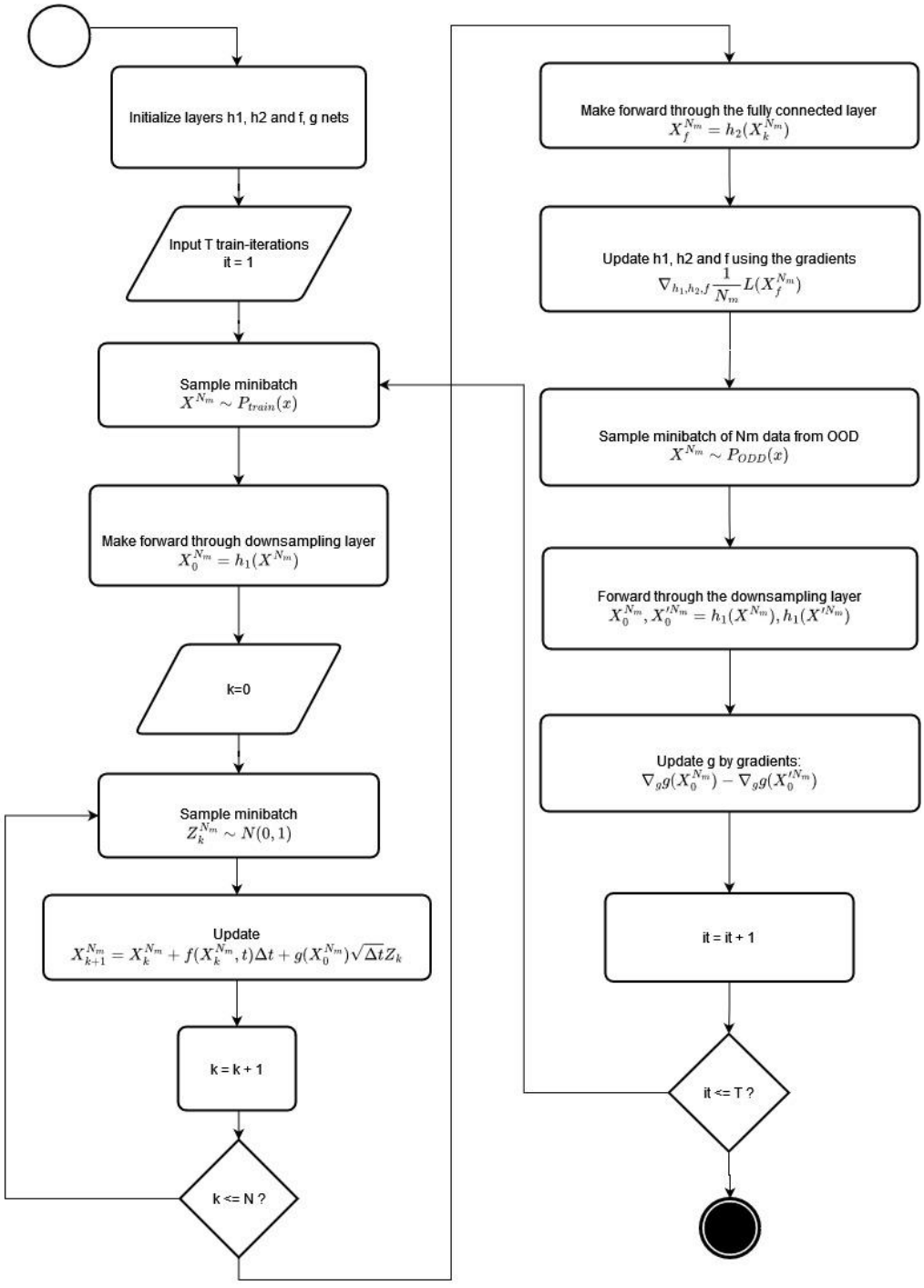
2.4. SDE-Net Training

Pseudocode for Training Algorithm of SDE-Net

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

Flux Diagram for Training Algorithm of SDE-Net



Objective Function

$$\min_{\theta_f} E_{x_0 \sim P_{train}} E(L(x_t)) + \min_{\theta_g} E_{x_0 \sim P_{train}} g(x_0, \theta_g) + \max_{\theta_g} E_{x_0' \sim POOD} g(x_0', \theta_g)$$

Adaptative step

$$x_{k+1} = x_k + f(x_k, t, \theta_f)\Delta t + g(x_0, \theta_g)\sqrt{\Delta t} Z_k, \text{ where } \Delta t = T / N$$

3. Experimental Results

3.1. Schema of experiments

Was realized the next experiments:

a. Out-of-distribution detection
b. Misclassification detection
c. Adversarial sample detection
d. Active learning

3.2. Description of the datasets

For the experiments, was used 2 datasets:

1) MNIST: The MNIST database for classification of handwritten digits has a training set of 60 000 examples and a test set of 10 000 examples.

2) SVHN: Street view house numbers database for classification contain over 600k labelled real-world images of house numbers taken from Google Street view. This dataset is structured with 26 032 digits for testing, 73 257 digits for training and 531 131 as extra training data.

3) Year Prediction MSD: regression dataset for prediction of the release year of a song from 90 audio features. Training set have 463 715 examples and test set 51 630.

3.3. Description of the graphics

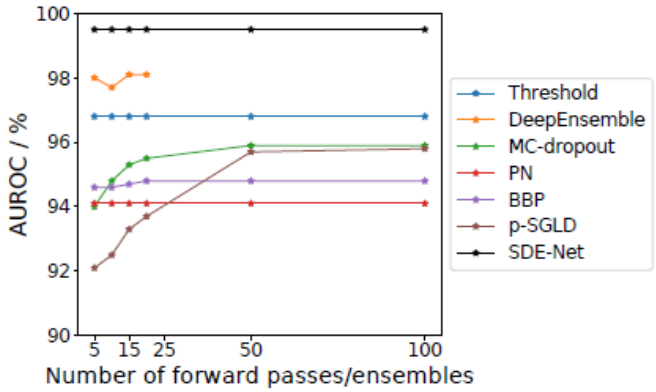


Figure 1. OOD Detection for classification

Effect number of forward passes/ ensembles on out-of-distribution (OOD) detection. Was used MNIST as ID data and SVHN as the OOD data. We can see the BNNs (MC-dropout, p-SGLD and BBP) requires more samples than SDE-Net to reach their peak performance at test time. In the case of the Deep Ensemble, the performance is saturated using 5 nets. Larger ensemble implies in losing of performance.

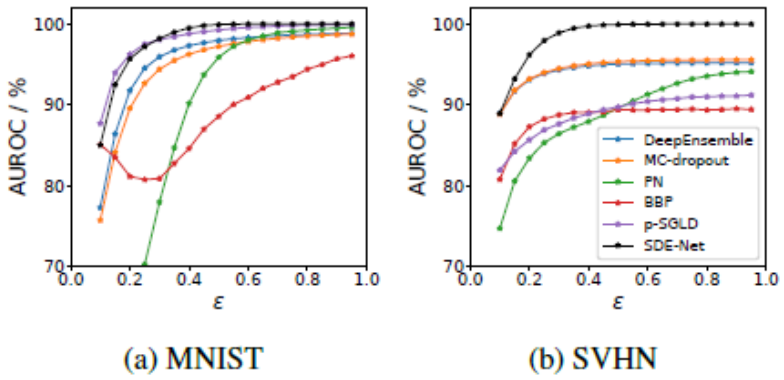


Figure 2. Performance of adversarial sample detection under FSGM attacks

Show the performance of different models when facing FGSM (Fast Gradient Sign Method) attacks. This experiment is important because FGSM highlight the vulnerability of machine learning models to adversarial examples through perturbations in the input data that is classified differently by the machine learning model than the original input was. In the experiment, we can see that SDE-Net is more robust model and resistant to this FGSM attacks in comparison with other models as the number of step size increases and increases the magnitude of the perturbation.

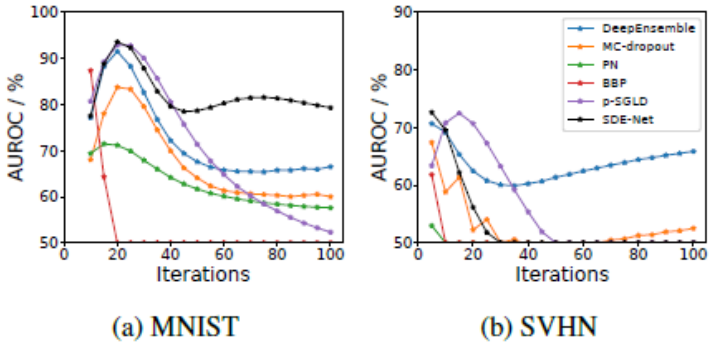


Figure 3. Performance of adversarial example detection under PGD attacks

Show the performance of different models when facing PGD (Projected Gradient Descent) attacks. This experiment considering the one of most powerful methods for generating robust adversarial examples. After multiple iterations of training under this PGD attack, for our models detect and classify correctly was much more difficult, and we can see good performance of SDE-Net in this scenario for MNIST dataset, but not enough in the case of SVHN, seeing a better performance of Deep Ensemble method and overconfidence in the SDE-Net and in the other models.

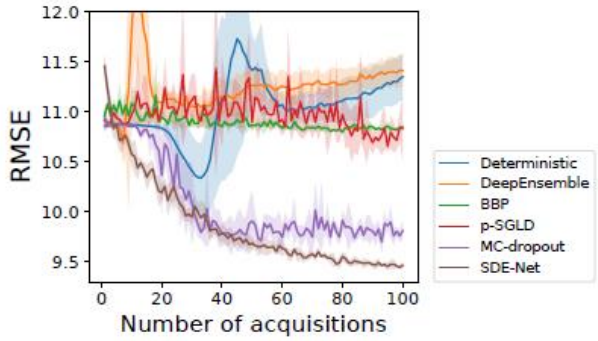
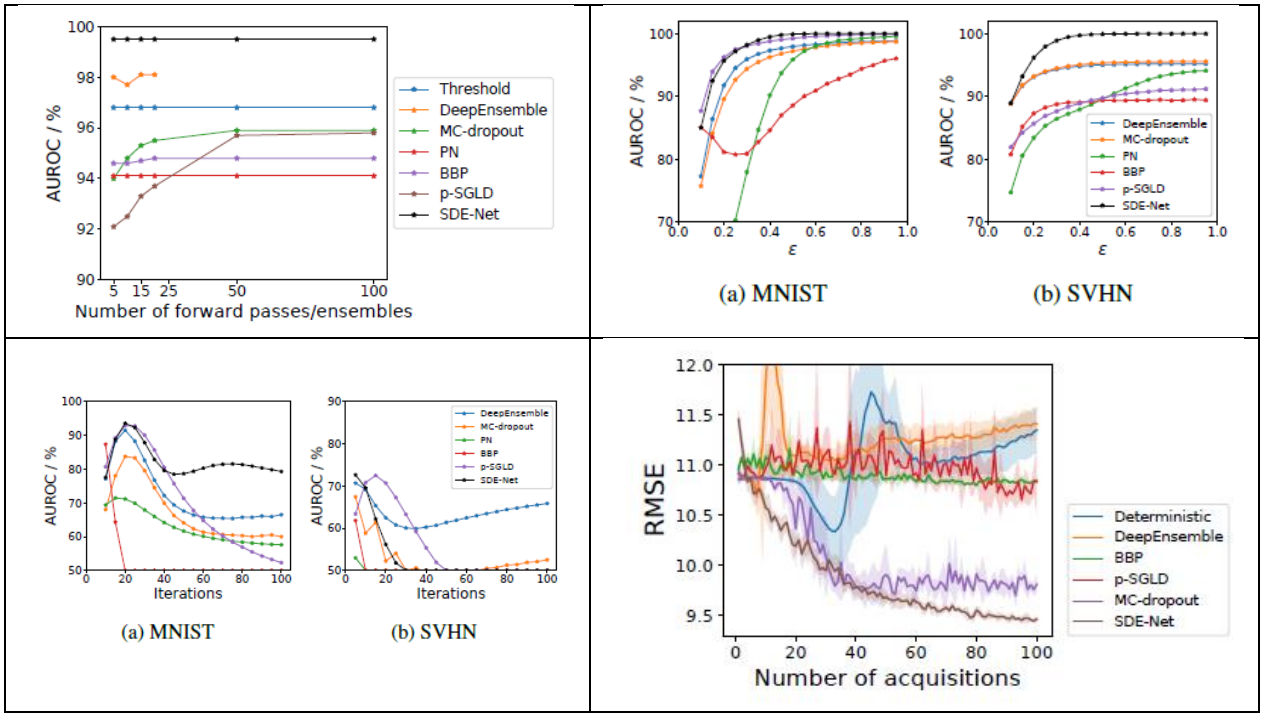


Figure 4. Performance of different models for active learning on Year Prediction MSD dataset

We can see in this active learning task, when much more labeled data the algorithm acquires, the RMSE (Root-Mean Square Error) consistently decreases and in the case of the SDE-Net, outperforms the other methods. Is interesting to note that the acquiring of new examples worsens the performance in the case of Deep Ensemble and Deterministic models, in others maintain and just the MC-dropout can achieve some comparative results respect SDE-Net.

3.4. Comparison between graphics



In the similarities between these experiments, we can see constantly that SDE-Net outperforms the other methods in the tasks where the uncertainty plays an important role. In other hand, the difference stays in each different task. In the first is focused in OOD detection, in the second in the reliable of the model under the FGSM attacks for adversarial sample detection, in the third, the same task but under PGD attacks, and in the last, the capacity of the algorithm improves its RMSE score when acquires new samples for training in the context of the regression task.

4. Software Tools

All the experiments were executed in GPU cluster tornado-k40 of Supercomputer Center of Polytech University.

The software tools used are:

- Programming language: Python in version 3.8.16
- Package manager: Anaconda 4.8 and pip 23.1.2
- Deep Learning framework: Pytorch
- Packages used: torch 2.0.1, torchvision 0.15.2, transformers 4.29.2

5. Execution of the Experiments

6. Discussion

With the implementation of the SDE-Net, was observed 3 main benefits:
(1) The aleatoric uncertainty and the epistemic uncertainty was explicitly separated in the model. The aleatoric uncertainty is naturally in the noise of data, and epistemic uncertainty was expressed by the introduction of the Brownian motion term in the model.
(2) It's not necessary define model priors and infer posterior probability distributions and it's straight-forward to implement.
(3) Available to use to classification and regression tasks.

7. Conclusion

(1) The model of SDE-Net can separate the different sources of uncertainty in comparison with existing non-Bayesian methods and in more straight-forward way in comparison with Bayesian methods.
(2) Experimentally was demonstrated that SDE-net outperforms state-of-art techniques the quantification of uncertainty and achieves robustness under different scenarios where uncertainty plays an important role.
(3) This research successfully established the connection between stochastic dynamical systems and neural networks for uncertainty quantification.
(4) In future directions, this approach has the potential to solve the problem of overconfidence under uncertainty scenarios in models which stay present in different deep learning applications.

8. References

[1] Kong, L., Sun, J., & Zhang, C. (2020). Sde-net: Equipping deep neural networks with uncertainty estimates. arXiv preprint arXiv:2008.10546.

[2] Blundell, C., Cornebise, J., Kavukcuoglu, K., & Wierstra, D. (2015, June). Weight uncertainty in neural network. In International conference on machine learning (pp. 1613-1622). PMLR.

[3] Hafner, D., Tran, D., Lillicrap, T., Irpan, A., & Davidson, J. (2020, August). Noise contrastive priors for functional uncertainty. In Uncertainty in Artificial Intelligence (pp. 905-914). PMLR.

[4] Lakshminarayanan, B., Pritzel, A., & Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles. Advances in neural information processing systems, 30.

[5] Chen, R. T., Rubanova, Y., Bettencourt, J., & Duvenaud, D. K. (2018). Neural ordinary differential equations. Advances in neural information processing systems, 31.

[6] PyTorch. (n.d.). PyTorch Documentation. Retrieved May 28, 2023, from https://pytorch.org/docs/stable/index.html

9. Annexes

Drift neural network

```
class Drift(nn.Module):

    def __init__(self, dim):
        super(Drift, self).__init__()
        self.norm1 = norm(dim)
        self.relu = nn.ReLU(inplace=True)
        self.conv1 = ConcatConv2d(dim, dim, 3, 1, 1)
        self.norm2 = norm(dim)
        self.conv2 = ConcatConv2d(dim, dim, 3, 1, 1)
        self.norm3 = norm(dim)

    def forward(self, t, x):
        out = self.norm1(x)
        out = self.relu(out)
        out = self.conv1(t, out)
        out = self.norm2(out)
        out = self.relu(out)
        out = self.conv2(t, out)
        out = self.norm3(out)
        return out
```