Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

R&D Project

Benchmarking Uncertainty Estimation Methods in Deep Learning for Regression

Aswinkumar Vijayananth

Submitted to Hochschule Bonn-Rhein-Sieg,
Department of Computer Science
in partial fulfillment of the requirements for the degree
of Master of Science in Autonomous Systems

Supervised by

Prof. Dr. Nico Hochgeschwender
Mr. Deebul Nair

January 2021

I, the undersigned below, declare that this work has not previously been submitted to this or any other university and that it is, unless otherwise stated, entirely my own work.

Date

Aswinkumar Vijayananth

Abstract

Deep Neural Network models are increasingly becoming a part of many Artificial Intelligence (AI) systems for safety-critical applications. However, their tendency to make over-confident and false predictions raises concerns on functional safety of their host systems. One of the ways in which the Deep Learning (DL) research community tries to solve this problem is by devising methods that estimate uncertainties associated with predictions of neural network models which are then utilized by the host AI systems to decide whether low-confidence decisions need further validation.

A considerable amount of research conducted on this topic has been attributed to methods that estimate uncertainties of neural network models for classification and only relatively fewer works focus on approaches for regression nets. There arises the need to enhance the existing or devise better uncertainty estimation methods suited for neural network models performing regression tasks. Also, there does not exist a comprehensive benchmark for such class of uncertainty estimation methods, which would reveal potential methods for the DL research community to conduct further research. This work compensates the need for such a benchmark which performs a detailed comparative evaluation of uncertainty estimation methods for regression networks.

In this work, state-of-the-art uncertainty estimation methods are carefully reviewed, analyzed and evaluated on the Udacity steering angle dataset and a group of three 1D datasets. The results show that "Deep Evidential Regression" (DER) outperforms other methods in terms of uncertainty estimation quality and prediction accuracy, while the method that combines "Monte-Carlo Dropout and Assumed Density Filtering" (MCDO_ADF) succeeds in providing an appropriate response to OOD and adversarially perturbed inputs.

Acknowledgements

I would like to thank my supervisors Prof. Dr. Nico Hochgeschwender and Mr. Deebul Nair for providing the opportunity to work on this interesting project.

I am grateful to Mr. Deebul for his continuous guidance, support and mentoring in every aspect of this research work which ensured its successful completion. Insights provided by Prof. Nico Hochgeschwender in the aspect of applying this work to safety-critical systems proved very useful.

I would like to extend my gratitude to Mr. Antonio Loquercio from ETH Zurich for sharing his work's source code which significantly improved the pace of this research work.

Finally, I would like to thank my parents for their love, support and guidance.

Contents

List of Figures

List of Tables

1

Introduction

1.1 Motivation

Deep Neural Network models are increasingly becoming a part of everyday software systems. This is due to their ability to exhibit state-of-the-art-performance in terms of accuracy, which often surpasses that of classical methods. Their increased adoption can also be attributed to the availability of high-performance computing hardware and software tools that enable easier expression and training of models. "As the barriers to building machine learning systems become lower, there is rising excitement around the idea of applying the technology in high-impact domains" [33] like medical diagnosis, autonomous driving and finance. However, the tendency of neural network models to output over-confident predictions [23] poses a limitation to their use in such areas. For instance, an overconfident incorrect prediction from a neural network model responsible for pedestrian detection in an autonomous car, can lead to an accident that costs human lives. Therefore, it is important to quantify the uncertainty linked to predictions of a neural network model, which could be in turn used by its host system to subject predictions with low-confidence to further validation or reject them (selective prediction). Apart from increasing the reliability of neural networks, estimating uncertainty can also be helpful in improving their model accuracies. For example, the performance of a neural network classification model that produces highly uncertain output for inputs belonging to a particular class, can be improved by retraining the network with more samples of the deficient class. [35] [39]

The Deep Learning research community has devised methods (for example [2], [10], [23]) to estimate uncertainty in neural network predictions, that enable models to output predictive distributions in place of point estimates and quantify uncertainty in terms of scale/variance of those distributions. An uncertainty estimation method is integrated to a neural network model by making some changes in its architecture, optimization process and/or inference procedure. Each method varies based on the nature of its target neural network's task and also by its approach to estimating uncertainty.

A significant portion of research conducted on the topic of uncertainty estimation has been attributed to methods for classification networks ([34], [27]) in contrast to regression. Even among literature on methods for regression nets, many focus on proving their technique's success over others, by comparing with a few selected methods on a limited number of aspects. Also, there does not exist a comprehensive benchmark for such a class of uncertainty estimation methods. This research work satisfies the need for such a

benchmark which would help the Deep Learning (DL) research community identify potential methods for further research and aid practitioners in choosing a suitable method based on their requirements.

1.2 Challenges and Difficulties

The following is a list of issues that need to be addressed by any uncertainty estimation method:

- An uncertainty estimation method is often integrated as a supplement to a deterministic neural net model to enable it output distributions in place of point estimates. Such an integration can have following consequences on the model:
  - Increased inference/test time
  - Increased model size
  - Major architectural changes that affect model's prediction accuracy
  - Changes in optimization process
- An uncertainty estimation method with many control parameters is not favored, as it adds to the burden of DL practitioners to tweak them based on the problem at hand.
- It is important for an uncertainty estimation method to rely more on data to model uncertainties than on assumptions about their underlying distributions.
- An alignment between estimated uncertainty levels and prediction error is expected. In other words, "a network should provide a calibrated confidence measure" [15].
- There does not exist a ground truth or a target for the value of uncertainty linked to a prediction. Therefore, it is important to devise a suitable strategy for evaluating uncertainty estimation methods.
- An uncertainty estimation method is commonly expected to produce relatively higher valued uncertainty estimates for Out-Of-Distribution (OOD) inputs than ones that fall within the training data distribution. However, this does not hold true practically for all cases. Invariance of a neural network model to changes in certain features that differentiate in and out of distribution inputs has a role to play. This has to be taken into consideration, while evaluating uncertainty estimation methods on OOD inputs.

1.3 Problem Statement

This research work aims to identify, evaluate and compare state-of-the-art uncertainty estimation methods for regression nets. Though uncertainty estimation methods for neural network models are broadly classified into Bayesian and non-Bayesian approaches, there exist other aspects that need to be considered before choosing a method for a diverse and a reasonable benchmark. The additional aspects are: number of forward passes required, need for architectural changes, need for changes in optimization, need for an ensemble and choice of distribution to model prior over weights. The chosen set of methods are carefully analyzed and two out of them are picked for an intensive evaluation.

As mentioned earlier, one of the predominant applications of uncertainty estimation methods is selective prediction in safety-critical systems. Therefore, it is prudent to evaluate the chosen pair of methods on a dataset related to a safety-critical application. In this research work, the Udacity steering angle dataset [37] is used for evaluating the methods. Apart from the steering angle dataset, a group of 1D datasets are also used to benchmark the methods of choice for the following benefits:

- Ease in visualizing results
- Improved control over parameters (related to both neural network training and uncertainty estimation methods)
- Creates possibilities to compare with baseline uncertainty estimation methods such as Gaussian Processes (GP)
- Reduced experimentation time

On both steering angle and 1D datasets, the chosen pair of uncertainty estimation methods are evaluated based on uncertainty estimation quality and prediction accuracy. However, when it comes to measuring the quality of uncertainty estimates, commonly used metrics such as Negative-Log-Likelihood (NLL) are often misleading [40]. As this work also uses NLL, the metric is carefully analyzed and its limitations are reported.

"A key use of uncertainty estimation is to understand when a model is faced with test samples that fall out-of-distribution (OOD) or when the model's output cannot be trusted" [2]. The last sections (Section 5.3.1, Section 5.3.2) of this work, evaluate the chosen pair of methods on appropriateness of their response to OOD inputs and adversarial attacks, which "pose a serious threat to the success of deep learning in practice" [1]. Such an evaluation is also crucial to assess the usefulness of an uncertainty estimation method in reporting high levels of uncertainty to the host system, when its corresponding NN model encounters a less familiar or an unknown input. To put everything concisely, this research work addresses the following research questions:

- **RQ1.** What are the existing state-of-the-art uncertainty estimation methods in Deep Learning for regression tasks?
- **RQ2.** How do the identified uncertainty estimation methods compare with each other and GP models, based on their uncertainty estimation quality and prediction accuracy of their respective models?
- **RQ3.** How do the identified uncertainty estimation methods compare with each other based on their response to OOD and adversarially perturbed inputs?

2

Background

This chapter provides a brief explanation about the two components of predictive uncertainty commonly considered by uncertainty estimation methods: epistemic and aleatoric.

2.1 Components of Predictive Uncertainty

2.1.1 Epistemic Uncertainty

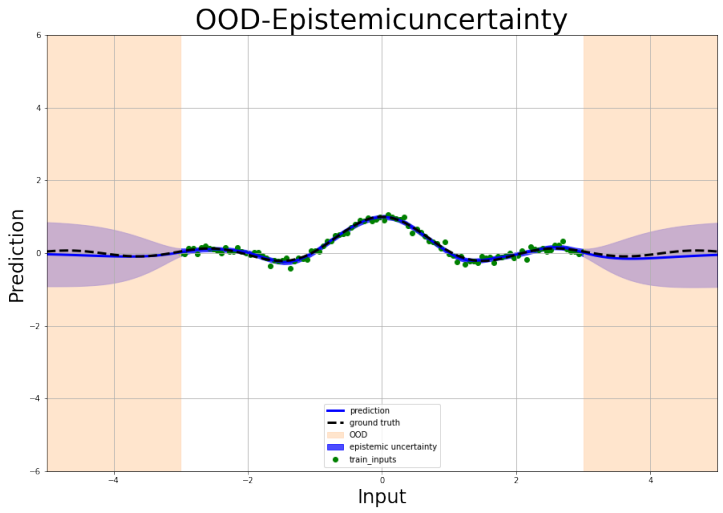


Figure 2.1: Plot depicting high levels of epistemic uncertainty associated with predictions for OOD inputs

Epistemic or Model uncertainty arises from the lack of inherent capacity of a neural net model to make predictions. The following contribute to model uncertainty:

- Dataset shift: Mismatch between training and test data distributions
- Structure uncertainty: Uncertainty in selecting the right model structure
- Uncertainty in selecting the right set of model parameters which best represents the observed data.

- Lack of knowledge in a model's portion about a given input.

Epistemic uncertainty is reducible in nature and can be explained away by training a given model with more data. Predictions for inputs from OOD region have a relatively higher value of model uncertainty linked to them than ones from within the region of training data.

2.1.2 Aleatoric Uncertainty

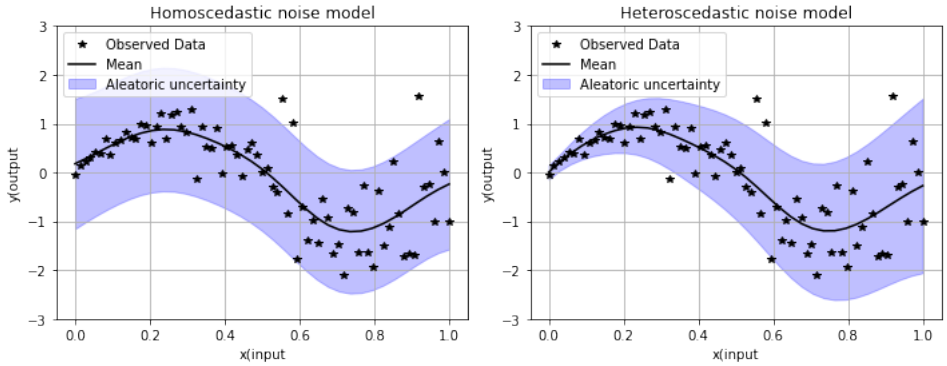


Figure 2.2: Plot depicting the difference between homoscedastic and heteroscedastic noise models

Aleatoric uncertainty (also called as data uncertainty) arises due to presence of noise in data. The following contribute to data uncertainty:

- Measurement imprecision
- Complex and multi-modal nature of data
- Digitization
- Artifacts induced from data pre-processing and compression techniques
- Random noise

Aleatoric uncertainty is considered to be irreducible as it cannot be compensated by training a given neural network model with more data. Unlike model uncertainty, aleatoric uncertainty levels do not increase abruptly for OOD inputs.

Based on the relationship between inputs and aleatoric uncertainty in a given problem, noise models can be categorized into two: homoscedastic and heteroscedastic. In a heteroscedastic model, aleatoric uncertainty is modeled as a function of inputs, while in the case of homoscedastic, a constant value of aleatoric uncertainty is considered. Plots in the Figure 2.2 show the difference between outputs of models with homoscedastic and heteroscedastic assumptions.

2.2 Relationship between Epistemic and Aleatoric Uncertainties

"The total uncertainty results from the combination of the model and data uncertainty." [26] and there also exists a relationship between them. Whenever aleatoric uncertainty linked to an input sample increases beyond a certain extent, it causes the sample to significantly deviate from its original structure resulting in increased epistemic uncertainty levels. Therefore, it is important for any uncertainty estimation method to not consider the pair of uncertainty components as mutually exclusive.

3

State of the Art

This chapter describes the set of state-of-the-art uncertainty estimation methods considered for this research work.

RQ1. What are the existing state-of-the-art uncertainty estimation methods in Deep Learning for regression tasks?

3.1 Dropout as Bayesian Approximation

The method proposed by Gal *et al.* [10] estimates model uncertainty in neural net models using Dropout [36] which is a commonly used regularization technique. The work proves the equivalence between a Dropout applied neural Network model and an approximated Deep Gaussian process model and establishes Dropout as a means to approximate the posterior predictive distribution. The absence of any need to make changes to the optimization process for integrating this method to a neural network model can be considered as one of its unique features. The technique applies to neural nets meant for both classification and regression tasks.

In order to obtain model uncertainty using this method in practice, dropout is enabled during the test-time and a given input is passed through the neural network a number of times equal to the pre-defined Monte-Carlo sample count hyper-parameter. The sample mean and variance of multiple stochastic forward passes correspond to the final model output and model variance respectively.

Though this method forms the basis of a few uncertainty estimation methods like [26], the need for multiple stochastic forward passes makes it computationally expensive and unsuited for real-time applications.

3.2 Deep Ensembles

Deep Ensembles [23] is a non-Bayesian approach to estimate predictive uncertainty by using outputs from an ensemble of neural networks. The work by LakshmiNarayanan *et al.* also proposes the idea of leveraging adversarial perturbations generated using the Fast Gradient SignMethod (FGSM) [14], to smoothen predictive distributions.

Authors show advantages of using Deep Ensembles over the base-line approach proposed by Gal *et al.* [10].The ability to report high uncertainty estimates for OOD and adversarial samples is claimed and also

proved by evaluating the method on a number of datasets for both regression and classification tasks. The work can be considered as one of the first frequentist approaches to estimate predictive uncertainty in neural networks.

With an increase in the number of models in an ensemble, the memory size increases as well. Therefore, the method is not suitable for very deep network architectures. Also, the need for diversity amongst ensemble members has a significant impact on the quality of estimated uncertainty.

3.3 Light-weight Probabilistic Deep Networks

Lightweight Probabilistic Deep Networks [11] by Gast *et al.* enables conversion of a deterministic neural network to its probabilistic equivalent which is capable of propagating and outputting parameters of the predictive distribution, in turn uncertainty linked to it. The conversion is achieved in two steps:

- Introducing a probabilistic output layer that produces moments of the predictive distribution as its outputs.
- Replacing intermediate activations with their probabilistic equivalents. Assumed Density Filtering (ADF) (explained in Section 3.7.2) a form of Expectation Propagation is used to achieve it.

The need for minor architectural changes and no change in the optimization process are major advantages of this method. The method does not represent weights probabilistically, which results in over-confident predictions.

3.4 Aleatoric Uncertainty as Learned Loss Attenuation

Kendall *et al.* proposed the method [17] which aims to include aleatoric variance parameter as a part of the loss function to learn it from training data. The work presents a framework that combines Monte-Carlo Dropout [10] and the heteroscedastic loss function to estimate model and data uncertainties respectively. The idea of learning a mapping from input data to aleatoric uncertainty is an important contribution of this work. The method disregards any relationship between components of uncertainty as they are treated as independent entities, which does not hold true.

3.5 Prior Networks

Malinin *et al.*[27] proposed a technique to estimate predictive uncertainties in neural nets for classification, by parameterizing a prior distribution over the predictive distribution. The work uses Dirichlet distribution as the higher-order (prior) distribution over predictive categorical distributions, due to existence of conjugate prior relationship between the pair which makes the posterior analytically tractable. In this way, the variances (uncertainties) around parameters of the categorical distribution are modeled. Parameters of the higher-order distribution are included as a part of the loss function which gets optimized.

Prior Networks can be considered as one of the earliest works whose method learns how to model uncertainty from the given training data. Another important contribution of this work is that the method separates out the uncertainty that arises due to mismatch between training and test data distributions as

"distributional uncertainty". Authors claim the work to outperform other uncertainty estimation methods when it comes to reporting misclassification and OOD input samples.

The proposed method is defined for classification setup and therefore cannot be used in regression nets. Also, the work lacks strong experimental evaluation as it is assessed only on toy datasets with two other uncertainty estimation methods.

3.6 Choice of methods for Intensive Evaluation

The methods listed so far suffer from at least one of the following downsides: increased model inference time, increased model size, under/over estimation of uncertainties , disregard any relationship between components of uncertainty. This research work considers "A General Framework for Uncertainty Estimation in Deep Learning" [26] and "Deep Evidential Regression" [2] for benchmarking, based on claims made by their authors that they overcome deficits of other methods,for an intensive experimental evaluation. The rest of this chapter explains both the methods descriptively.

3.7 A General Framework for Uncertainty Estimation in Deep Learning

3.7.1 Overview

This work proposes a technique to distinctively estimate data and model uncertainties associated with an output of any neural network model. The technique is here after referred to as "MCDO_ADF", representing the fact that is a combination of two ideas, Monte-Carlo Drop Out (MCDO) and Assumed-Density-Filtering (ADF). MCDO_ADF treats the two uncertainty components to be related, which sets it apart from most of other uncertainty estimation methods that treat them to be independent. The method employs Bayesian Belief Networks (BBN) combined with Monte-Carlo sampling for estimating the model uncertainty and relies on the idea of Assumed Density Filtering for estimating data uncertainty associated with an output.

Authors of the MCDO_ADF technique claim it to be a general framework to estimate uncertainties in neural networks. They give following reasons to validate their claim:

- Using this uncertainty estimation method does not require any architectural changes in the target neural network.
- Applicability of the method to neural network models of different tasks.
- Absence of any need of make changes in the optimization process.
- Ability of the technique to be applied to already trained models.

The upcoming sections of this chapter explain the MCDO_ADF technique and also analyze its claimed "generality" by using it in a Resnet8 based neural network regression model meant for the application of steering-angle prediction in autonomous cars. (Note: A detailed description of the data set, training and inference procedures of the neural network model is available in Chapter 4).

3.7.2 Integrating MCDO_ADF into a Neural Network and Estimating Uncertainties

MCDO_ADF as an Algorithm

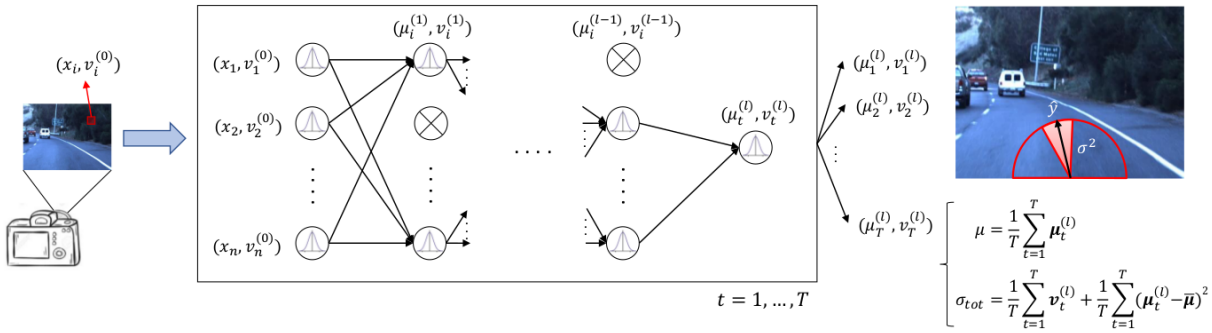


Figure 3.1: Illustration of the MCDO_ADF technique.Here x_i denotes the input through the i^{th} unit of a given hidden layer,$x_{i(n)}$ denotes the noise variance input to the ith unit of the nth layer. Circles with crosses inscribed denote the dropped out neurons whereas the ones with the Gaussian distribution symbol denote the active units. T values of μ and v are collected from T stochastic forward passes. Image source: [26] (p.4)

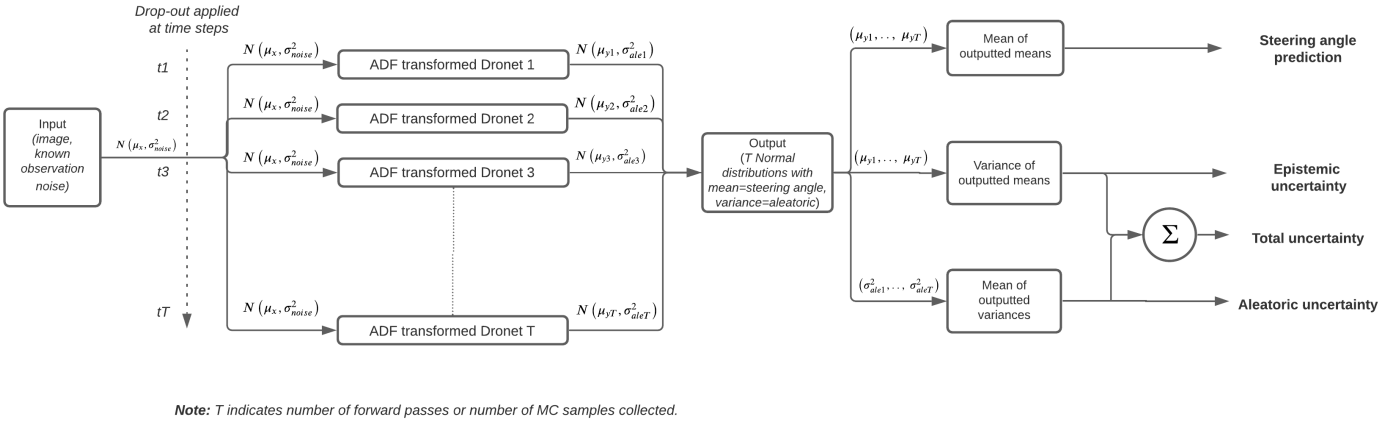


Figure 3.2: Block diagram depicting inference procedure using MCDO_ADF in Dronet

Estimating uncertainty using the MCDO can be formulated as an algorithm consisting of the following steps:

- Transform the neural network of interest to its ADF (Assumed Density Filtering) equivalent.
- Collect a predefined number (T) of Monte-Carlo (MC) samples by forwarding inputs and noise variances (x, v)) stochastically through the network for T times.
- Computation of output predictions and uncertainties

Assumed Density Filtering (ADF)

The MCDO_ADF technique considers sensor noise to be the primary source of data uncertainty in neural network predictions and therefore feeds it to the neural network model during the inference. In order to propagate the input data distribution (parameterized by the input as its mean and sensor noise as variance) ADF is used. Briefly in the context of MCDO_ADF, ADF replaces every input activation into a probability distribution and also approximates the same using a tractable Gaussian distribution and makes both the mean and noise variance available in the output layer.Following points describe Assume Density Filtering in a more detailed manner:

- Assumed Density Filtering (ADF) is a technique in Bayesian machine learning to approximate intractable and complex distribution with distributions that are easy to handle. In the case of Bayesian Inference, ADF aims to project the true posterior onto a distribution of choice. The exponential family of distributions are a popular choice.
- In the case of MCDO_ADF there is a need to propagate the input data distribution so that the values of its mean and variance (noise variance) are available in the output layer.
- The input data distribution is considered to be Gaussian in nature. Every intermediate layer outputs the transformed version of the input distribution. However, when it propagates through non-linearities in a neural network the resulting distribution need not be essentially another Gaussian. Such a distribution emerging out of non-linear blocks is also conditioned by distribution over activations of the preceeding layers. Therefore, the resulting distribution becomes intractable.
- Such intractable and complex distributions are estimated using ADF by:
  - Assuming conditional independence between distribution outputted from a given layer with its preceeding layers.
  - Approximating the complex distribution with a Gaussian distribution whose pair has the least possible value of Kullback-Leibler divergence. ADF achieves this by matching the first two moments of the distributions.
  - In practice, this is achieved by optimizing a global variational objective.

In practice, every building block of a neural network has its corresponding ADF version and therefore during the inference time the entire model has to be transformed to its ADF equivalent. This gives the ability to the neural network model to propagate and output distributions which represent data uncertainty.

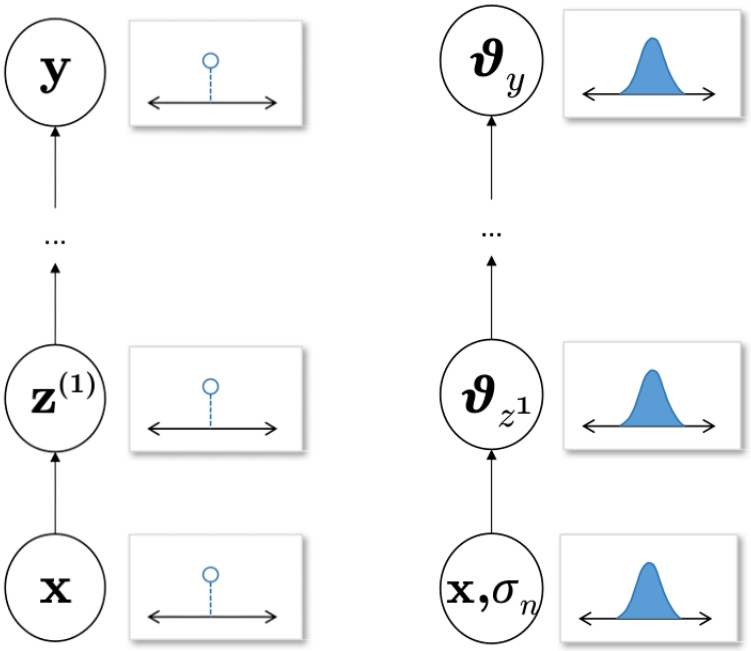


Figure 3.3: Illustration of forward passes in deterministic and ADF versions of a neural network. Here x denotes the input activations,$z^{(n)}$ denotes activation input to the nth layer,y denotes the output, θ_{z^n} represents input activation to the nth layer expressed as a probability distribution and σ corresponds to the noise variance. Image source: [26](p.1)

Data Uncertainty Estimation

The ADF transformed neural network produces two outputs from the final layer: mean ($\mu_{t^{(l)}}$) and variance ($v_{t^{(l)}}$) of the propagated distribution, as shown in the Figure 3.1. The pair of values is outputted for each of the T stochastic forward passes (described in the next paragraph) and the mean of T variance values is considered to be the value of data uncertainty. Likewise, the mean of T predictions is considered to be the model's prediction for the given input.

$$\text{prediction} = \mu = \frac{1}{T}\sum_{t=1}^{T}\mu_{t^{(l)}} \tag{3.1}$$

$$\text{data uncertainty} = \sigma_{data} = \frac{1}{T}\sum_{t=1}^{T}v_{t^{(l)}} \tag{3.2}$$

Monte-Carlo Dropout (MCDO)

This uncertainty estimation method relies on the idea of Monte-Carlo (MC) sampling to estimate model uncertainty associated any prediction. In practice, MC sampling is achieved by enabling dropout during

the test time and obtaining the desired number of samples (T), which are nothing but outputs of the neural network model during different forward passes of the input. Enabling dropout introduces stochasticity during those forward passes.

Following points briefly describe the dropout technique in a general context:

- Dropout [36] is primarily a regularization technique used while training neural networks in order to avoid over-fitting.
- During dropout certain nodes of a given neural network layer are not considered for training. The nodes are ignored with a probability equal to the dropout rate (often denoted by **p**).
- Using dropout during training makes neural network layers to adapt in such a way that they cope with mistakes made by the prior layers.

In the context of Bayesian inference, the Dropout technique is used to approximate the posterior distribution over weights of a given neural network when the training data and labels are given ($P(W|X,Y)$). The approximation is obtained by applying dropout at the test-time. This makes it possible to obtain multiple predictions for any given input from different architectures resulting from application of dropout to the base neural network model. The different architectures obtained along with their weights can be considered as Monte-Carlo(MC) samples from the space of all possible architectures. The number of MC samples to be obtained is a hyper-parameter and denoted by T. In another perspective, T equals the number of forward passes through different architectures with different sets of weights $\{W_1^t, .., W_L^t\}_{t=1}^T$ (L) denotes the number of dropout applied layers in the neural network). The first and second moments (mean and variance) of predictions obtained from these stochastic forward passes of given input are utilized to compute model uncertainty. One of the highlights of this technique is that its usage does not require any architectural changes and also can applied to already trained neural net models. The hyper-parameters T and p significantly impact the effectiveness of this technique. In the case of p a very high value (close to 1) increases sparsity in nodes and also results in longer convergence-time while a low value eliminates the MC-sampling utility. For our experiment, the value of $p = 0.02$ is used. The hyper-parameter T significantly impacts the inference time of a neural network model and therefore has to be chosen optimally based on the run-time requirement of the system where the model would be deployed.

Model Uncertainty Estimation

The MCDO_ADF technique estimates model uncertainty using predictions generated from the neural network model during multiple (T) forward passes, while the dropout is enabled. A given input is processed by the model T times, with a new combination of neurons considered for almost every forward pass. This produces an effect of gathering predictions from an ensemble consisting of T neural network models with different architectures. The variance of T gathered predictions is the estimated model uncertainty. In the following equations $\mu_{t^{(l)}}$ signifies the mean output from the ADF transformed version of the Model during T^{th} forward pass.

$$\text{model uncertainty} = \sigma_{model} = \frac{1}{T}\sum_{t=1}^{T}(\mu_{t^{(l)}} - \overline{\mu})^2 \tag{3.3}$$

$$where, \overline{\mu} = \frac{1}{T}\sum_{t=1}^{T}(\mu_{t^{(l)}}) \tag{3.4}$$

Combining ADF and MCDO

The MCDO_ADF method considers a relationship to exist between the two components of uncertainty (data and model). The relationship is realized in this technique by combining both the ideas of ADF and MCDO. During inference,

- The given neural network model is transformed to its ADF equivalent so that the output layer produces both predictions (mean) and noise variance as the model's final outputs.

- For estimating model uncertainty, dropout is enabled in the ADF transformed version of the original neural network following which T MC samples are collected during T stochastic forward passes. It is this application of dropout on the ADF transformed version that produces the "effect of ensembling T ADF neural networks" and also considers any relationship between the two uncertainty components.

- Combining ADF and MCDO leads to another intuitive realization about the uncertainty components in this setup. Even when a particular input fed to the neural net model was observed frequently during training, if corrupted due to sensor noise then it will have high values of both data and model uncertainties.

Total Uncertainty

The predictive uncertainty is estimated by summing up both its components (data and model) and is given by the following equation.

$$\text{predictive uncertainty} = \sigma_{total} = \frac{1}{T}\sum_{t=1}^{T}((\mu_{t^{(l)}} - \overline{\mu})^2 + v_{t^{(l)}}) \tag{3.5}$$

In summary, both ADF and MCDO techniques approximate probability distributions of data and model with Normal distributions respectively. ADF propagates the input data distribution and approximates it as Gaussians in every neural network layer while MCDO approximates the distribution around weights by sampling and forms a Gaussian distribution out of the samples. The variances of these Gaussian distributions are considered to be the uncertainty components and are summed up to yield the predictive uncertainty.

3.7.3 Inference Procedure

The MCDO_ADF method can be applied to already trained deterministic version of the neural network models as mentioned in the Section 3.7.1. However, it is also possible to train the neural network model of interest with dropout enabled and use the same for inference. In order to estimate the predictive uncertainty during inference:

- Every layer of the neural network has to replaced with its ADF equivalent so that they are equipped with the ability to propagate data distributions. The implementation of ADF equivalents for most of the neural network building blocks is available in the Github repository of [11].
- The value of noise variance (a constant value) obtained from the sensor's data sheet is fed along with the input data to the ADF transformed input layer of the network. During propagation through intermediate layers, it is ensured that at least a minimum value of variance is propagated. In the case of experiment described in Chapter 4, a minimum value of 0.001 is used.
- Every input along with the noise variance undergoes T stochastic forward passes through the network to generate T predictions.

The experiment described in the next chapter discusses more on practical aspects of this technique.

3.7.4 Downsides

- The need for multiple (T) forward passes to obtain MC samples is computationally expensive and cannot be afforded in the case of real-time systems. While there is an option to reduce the value of T, it increases the difference between approximated and underlying distribution over weights of the model, thereby affecting the method's performance.
- The method considers sensor noise to be the only source of data uncertainty. Also, it treats the noise to be additive Gaussian in nature. However, sensor noise is just one of the factors contributing to data uncertainty. For instance, in the case of image data, usage of a lossy compression technique can also contribute to its noise. Also, it is possible for a given sensor to produce data whose noise levels differ. As it is impossible to consider and model every possible noise source, it is important for an uncertainty estimation method to learn to differentiate noise and useful information from given data.
- The authors of MCDO_ADF quote its ability to be applied to already trained models as one of the key reasons for its generality. However, retraining a neural network model is something feasible in most of the cases.
- As hyper parameters such as drop-out rate (p), number of MC samples (T) and noise variance have a major role to play in this technique, it adds to the responsibilities of the practitioner to optimally choose them based on the problem at hand.

3.8 Deep Evidential Regression

3.8.1 Overview

Deep Evidential Regression proposes a method (hereafter referred to as "DER" to estimate predictive uncertainty primarily in neural networks for regression, by simultaneously learning a hierarchy of distributions. The learned hierarchy consists of two levels of distributions: 1. A lower level Gaussian likelihood distribution over data, with parameters (mean μ and variance σ^2) 2. A higher level (also called Evidential) Normal-Inverse Gamma distribution over the parameters of the lower level distribution. In the perspective of the Bayesian Inference, the higher-order distribution can be taken as a prior over the lower-order distribution which is obtained by evaluating likelihood of known data points for a particular choice of μ and σ^2. The evidential distribution evaluated at any particular instance (a combination of μ and σ^2), provides the subjective belief mass of the corresponding lower-order distribution there. This subjective belief mass is also called as "evidence". Lack of evidence means existence of uncertainty and therefore the value of evidence is used to quantify predictive uncertainty.

In order to put the above mentioned ideas into practice, DER provides a loss function whose objectives are to:

- Fit the training data to the evidential model.
- Learn the evidential prior which would provide uncertainty estimates during inference.
- In simple words, to learn the parameters of the higher-order evidential distribution.

The upcoming sections of this chapter explain the method in a detailed manner.

3.8.2 Conjugate Priors

Let us consider a learning problem where Random Variables (RV) Θ and Y represent model parameters and data respectively. Assuming that RVs are jointly distributed and applying Bayes Rule to determine the probability distribution of Θ given Y,

$$P(\Theta|Y) = \frac{P(Y|\Theta)P(\Theta)}{P(Y)}$$

The equation can be expressed in words as follows:

$$\text{posterior distribution of } \Theta \text{ given Y} = \frac{\text{likelihood of Y given } \Theta \text{ . prior over } \Theta}{\text{marginal Likelihood of Y}}$$

During inference, for a particular choice of functions to represent the likelihood distribution, the nature of prior distribution function matches the nature of posterior distribution function. For example, if a normal distribution with unknown mean and variance is used to represent the likelihood distribution and if a Normal-Inverse Gamma distribution (NIG) (described in the next subsection) is used to represent the prior

distribution then the nature of posterior probability distribution is also observed to be Normal-Inverse Gamma in nature. This can be briefly written as "Normal-Inverse Gamma distribution is the conjugate prior for Normal distribution in likelihood". Conjugate priors help to reduce computations involved in determining the $P(\Theta|Y)$ value every time during the process of determining optimal set of parameters. Beta, Gamma and Normal distributions are favorite choices for priors as they act as conjugate priors for different likelihood distribution functions. In the context of DER, the conjugate prior relationship between distributions is used to introduce a hierarchy between them in order to probablistically model the likelihood distribution.

Distribution Hierarchy

Let us assume using a Normal distribution $\mathcal{N}(\mu, \sigma^2)$ to model a set of data points $x_1, x_2, .., x_i$."When a probability distribution A is used to model the given set of data, the uncertainty in the fit is described by probability distribution/s B over parameters of A". This means defining probability distributions over the set of parameters μ, σ^2 helps in describing uncertainty in the model fit.

The probability distribution of μ is modeled by a normal distribution due to its Gaussian nature and the fact that $\mu \in \mathbb{R}$. On the other hand, a Gamma distribution ($\Gamma(\alpha, \beta)$) is used to model the probability distribution of σ^2 owing to its strictly positive nature. The following figure illustrates hierarchical relationship between distributions under consideration, where $(\mu_0, \sigma_0^2), (\alpha, \beta)$ represent the parameters of the higher-order Normal and Gamma distributions respectively.

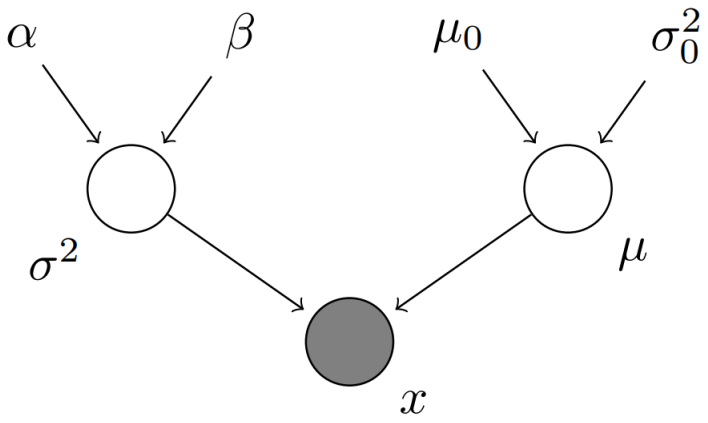


Figure 3.4: Hierarchy in distribution parameters.Image source: [16] (p.10)

Alternatively, a Normal-Inverse-Gamma distribution (often represented as NIG (α,β,γ,λ)) can be used to model the probability distribution of μ and σ^2 jointly. DER uses the distribution to realize its objectives. Significance of NIG's parameters is explained in the next section.

3.8.3 Evidential Distribution

From the Perspective of Bayesian Inference

Let us consider a regression problem with an available dataset D with N pairs of data labels and targets represented by $(x_1, y_1), .., (x_N, y_N)$. The DER method assumes that the targets are drawn independent and identically from a Gaussian distribution with unknown mean and variance represented by μ and σ^2 respectively.

$$(y_1, .., y_N) \sim \mathcal{N}(\mu, \sigma^2) \tag{3.6}$$

The parameters μ and σ^2 are considered to be random variables that follow Gaussian and Inverse-Gamma distributions respectively.

$$\mu \sim \mathcal{N}(\gamma, \sigma^2 \lambda^{-1}) \tag{3.7}$$

$$\sigma^2 \sim \Gamma^{-1}(\alpha, \beta) \tag{3.8}$$

where $\alpha, \beta, \gamma, \lambda$ denote parameters of the higher-order Normal Inverse Gamma (NIG) distribution. Let $\theta = (\mu, \sigma^2)$ denote the parameters of one instance of Gaussian distribution generating targets y_i and $m = (\alpha, \beta, \gamma, \lambda)$ denote the set of NIG distribution parameters.

We are interested to model the distribution around θ. Applying Bayes Rule, we get

$$P(\theta|m) = \frac{P(m|\theta)P(\theta)}{P(m)}, \tag{3.9}$$

$$\text{posterior. dist. over } \theta \text{ for the given value of m} = \frac{\text{likelihood of m evaluated at the given value of } \theta \text{ x prior over } \theta}{\text{likelihood of m evaluated at all possible values of } \theta}$$

Here, the prior over θ is a NIG distribution and the likelihood function is Gaussian in nature. Therefore, the posterior takes the form of an NIG distribution expressed as follows:

$$P(\mu, \sigma^2 \mid \gamma, \lambda, \alpha, \beta) = \frac{\sqrt{\lambda}}{\sigma\sqrt{2\pi}} \frac{\beta^\alpha}{\Gamma(\alpha)} \left(\frac{1}{\sigma^2}\right)^{\alpha+1} \exp\left(-\frac{2\beta + \lambda(\gamma - \mu)^2}{2\sigma^2}\right) \tag{3.10}$$

Significance of NIG Parameters

γ and α are shape and location parameters of NIG distribution respectively. β refers to the inverse-scale (rate) parameter. This means that spread of the distribution is inversely related to β. There is a relationship that exists between parameters of the NIG distribution. γ can be interpreted as the sample mean of λ virtual observations, determining NIG's location. On the other hand, spread of the NIG distribution can be considered to have calculated from 2α virtual observations whose sample mean equals γ and their squared deviations summing to 2β. DER considers the count of virtual observations as evidence

(ϕ) in support of the data sample at hand.

$$\phi = \lambda + 2\alpha \tag{3.11}$$

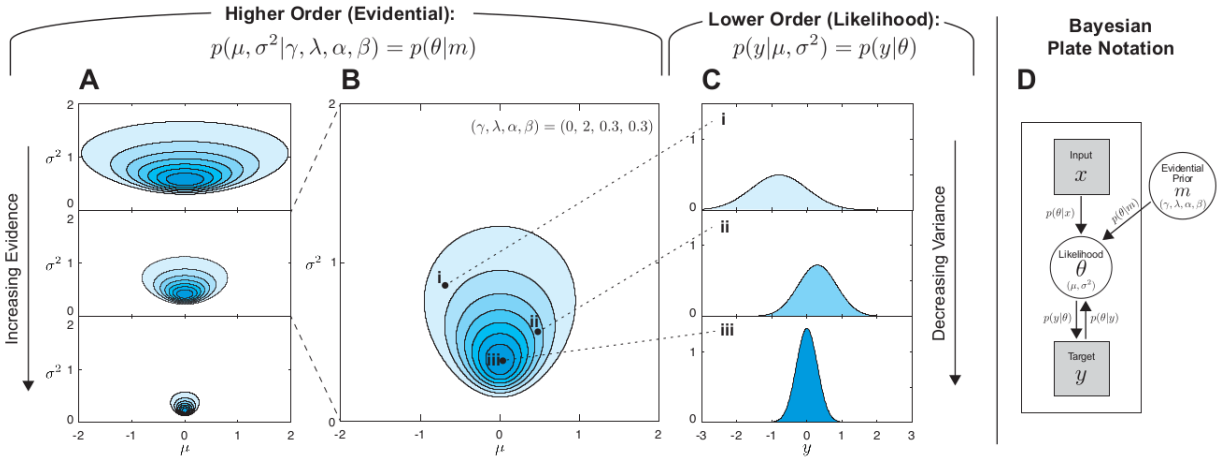


Figure 3.5: Realizations of the NIG distribution. Image source: [2](p.3)

Figure 3.5 illustrates the impact of increase in evidence on the shape and spread of the NIG distribution (column **A**) and various realizations of the lower-order likelihood distribution (column **C**) from a given instance of NIG distribution (column **B**). Following are some of the key insights that can be obtained from the illustration:

- With increase in evidence (as expressed in Equation 3.11) the belief mass increasingly concentrates around a specific value pair of μ and σ^2 in the column **A** meaning a reduction in uncertainty.
- Column **B** illustrates the evidential distribution centered around a particular value pair of μ and σ^2. This intuitively means that every point on the distribution corresponds to parameters of a possible likelihood distribution.
- Sampling the higher order distribution at various locations yield likelihood distributions of varying levels of evidence associated with them. Column **C** in the illustration shows such realizations. Darker the blue shade used to represent a likelihood-distribution, higher the evidence level associated with it.

3.8.4 Evidential Learning Objectives

As described in the overview (Section 3.8.1), the objectives of DER are two fold: 1. Maximize the model fit and 2. Minimize the evidence measure in an event of error. The method realizes these objectives

in form of a loss function/s (two forms) which is integrated to the neural network of choice and optimized during training.

Maximizing the Model Fit

This objective of DER focuses on learning the underlying patterns in the data and also increasing the belief mass/evidence in favor of right predictions.

Rewriting the Equation 3.9

$$P(\theta|m) = \frac{P(m|\theta)P(\theta)}{P(m)}$$

Let us assume that we observe our target y_i which when added to the above equation yields,

$$P(\theta|y_i, m) = \frac{P(y_i|\theta, m)P(\theta|m)}{P(y_i|m)} \tag{3.12}$$

Before interpreting the above equation it is important to recollect the fact that in Bayesian Inference every Random Variable (RV) involved is considered to be jointly distributed. In the case of above equation, Random Variables Y,Θ and M corresponding to y_i, θ and m are jointly distributed.

In the above equation, we determine the probability distribution around θ when it is conditioned under specific values of y_i and m. The two terms in the numerator denote likelihood and prior as described in Section 3.8.3. The denominator term is termed as "marginal likelihood" or "evidence" in Bayesian inference, as it yields the total probability mass of $Y = y_i$ for all possible realizations of θ in the model parameterized by m. The evidence term is also important to normalize the likelihood so that it represents a probability measure. Column D of Figure 3.5 illustrates Bayesian inference in DER. The marginal likelihood term can be represented mathematically as follows:

$$P(y_i|m) = \int_{\theta} P(y_i|\theta, m)P(\theta|m)d\theta \tag{3.13}$$

$$P(y_i|m) = \int_{\sigma^2=0}^{\infty}\int_{\mu=-\infty}^{\infty} P(y_i|\mu, \sigma^2)P(\mu, \sigma^2|m)d\mu d\sigma^2 \tag{3.14}$$

The proposed loss function aims to determine the set of parameters m which maximizes the term $P(y_i|m)$(evidence) for the given target y_i. Similar to Maximum Likelihood Estimation, the objective of maximization of marginal likelihood is re-framed as minimization of negative log of marginal-likelihood (NLL) for computational convenience. The loss function is expressed as:

$$\mathcal{L}_i^{NLL}(w) = -logP(y_i|m) \tag{3.15}$$

$$\mathcal{L}_i^{NLL}(w) = -log(2^{0.5+\alpha}\beta^{\alpha}\sqrt{\frac{\lambda}{2\pi(1+\lambda)}}(2\beta + \frac{\lambda(\gamma - y_i)^2}{1+\lambda})^{-0.5-\alpha}) \tag{3.16}$$

Alternative to the usual way of minimizing Negative-Log-Likelihood (NLL) the author proposes yet another form for the loss function which minimizes sum-of-squared errors between the prior and data sampled from the likelihood function. Following is the expression for the Sum Of Squared errors (SOS)

version of the loss function:

$$\mathcal{L}_i^{SOS}(w) = (\frac{\Gamma(\alpha - 0.5)}{4\Gamma(\alpha)\lambda\sqrt{\beta}})(2\beta(1+\lambda) + (2\alpha - 1)\lambda(y_i - \gamma)^2) \tag{3.17}$$

The author claims the SOS version of loss function to be relatively stable while training and also to perform better than the other during evaluation. The portion of loss function $\mathcal{L}$ described in this section only achieves the "model fitting" objective of DER.

Minimizing Evidence on Errors

The second objective of DER aims to minimize the evidence measure or to inflate uncertainty in the absence of training data. DER expresses this objective by adding a regularizer term to the loss function which penalizes the loss function in an event of its prediction deviating from the ground truth label. The penalty is scaled by evidence which expressed as sum of virtual observations as described in Equation 3.11. Following is the expression for the regularizer term,

$$\mathcal{L}_i^R(w) = \|y_i - \gamma\| .(2\alpha + \lambda) \tag{3.18}$$

Here p refers to the order of norm used to represent the difference between the ground truth label y_i and predicted mean γ. Author uses the value of p=1 claiming it to be stable during the training process.

Putting both its objectives together the evidential loss function can be expressed as:

$$\mathcal{L}_i(w) = \mathcal{L}_i^{SOS}(w) + \mathcal{L}_i^R(w) \tag{3.19}$$

3.8.5 Estimating Uncertainty

Epistemic uncertainty which quantifies the model's inherent lack of knowledge associated with an output can be expressed as the variance around its predictions.

$$Var(\mu) = \frac{\beta}{(\alpha - 1)\lambda} \tag{3.20}$$

The λ term in the denominator refers to the number of virtual observations. Aleatoric uncertainty can be computed with the following expression:

$$\mathbb{E}[\sigma^2] = \frac{\beta}{\alpha - 1} \tag{3.21}$$

From equations 3.21 and 3.20 both components of uncertainty can be related as follows:

$$\text{epistemic uncertainty} = \frac{\text{aleatoric uncertainty}}{\lambda} \tag{3.22}$$

This means that the epistemic uncertainty component is the mean of aleatoric uncertainty over λ virtual observations.

Predictive uncertainty can be evaluated as the sum of epistemic and aleatoric uncertainty components.

$$\text{predictive uncertainty} = \text{aleatoric uncertainty} + \text{epistemic uncertainty} \tag{3.23}$$

From eqns 3.21 and 3.20 predictive uncertainty can be computed as follows:

$$\text{predictive uncertainty} = \frac{\beta}{\alpha - 1} + \frac{\beta}{(\alpha - 1)\lambda} \tag{3.24}$$

After simplification,

$$\text{predictive uncertainty} = \frac{\beta(1 + \lambda)}{(a - 1)\lambda} \tag{3.25}$$

4

Methodology

This chapter describes datasets, neural network and Gaussian Process (GP) models used for the experiments conducted in this research work.

4.1 Datasets

This research work evaluates the considered pair of uncertainty estimation methods on two datasets: Udacity steering angle dataset and a set of 1D datasets, whose descriptions are given in the following sections.

4.1.1 Steering Angle Dataset

The Udacity steering angle dataset (available in [37]) consists of driving scene images captured by a set of three cameras (left, center, right) mounted behind the windshield of an ego vehicle. Along with the captured images, the dataset also contains steering angle, torque and vehicle speed values logged at that particular instance. Experiments conducted in this research work only utilize the set of images captured by the center camera and their corresponding steering angles expressed in radians. The data set contains driving scene images captured during different weather and traffic conditions (dataset samples can be found in the Figure 4.1). The data set consists of 33,808 images in total and for experiments conducted in this work, a train-validation-test split ratio of 80:5:15 is used. The following table gives further details about the dataset.

Dataset folder identifier	**Conditions**	**Count**			
		Train	**Validation**	**Test**	**Total**
HMB_1	Divided highway and sunny conditions	3521	220	660	4401
HMB_2	Two lane road and sunny conditions	12637	790	2369	15796
HMB_4	Divided highway segment	1579	99	296	1974
HMB_5	Guard rail and two lane road	3388	212	635	4235
HMB_6	Divided multi-lane highway with a fair traffic and shadows prevalent all over	5922	370	1110	7402

Table 4.1: Train-validation-test split of the Udacity steering angle dataset

Figure 4.1: Sample images from the Udacity steering angle dataset. Image source: Udacity steering angle dataset [37]

Steering angle prediction is both a safety and a time critical application which serves as an essential component of any autonomous vehicle. As enhancing functional safety in such applications is one of the key objectives of using uncertainty estimation methods, the steering angle data set is chosen for benchmarking the techniques considered for this research work.

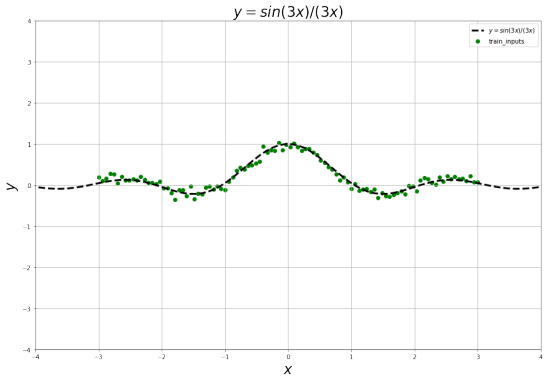
4.1.2 1D Datasets

Identifier	**1D function** $y = f(x) + \epsilon$	**Noise parameters** $\epsilon \sim \mathcal{N}(\mu, (\sigma)^2)$	**Train and test data range (points are equally spaced)**
fn_1	$y = sin(3x)/3x + \epsilon$	$\epsilon \sim \mathcal{N}(0, (0.08)^2)$	train: 100 points in [-3,3], test: 100 points in [-5,5]
fn_2	$y = 0.1x^3 + \epsilon$	$\epsilon \sim \mathcal{N}(0, (0.025)^2)$	train: 50 points in [-4,-1] $\bigcup$ 50 points in [1,4] test: 200 points in [-4,4]
fn_3	$y = -(1 + x)sin(1.2x) + \epsilon$	$\epsilon \sim \mathcal{N}(0, (0.04)^2)$	train: 50 points in [-6,-2] $\bigcup$ 50 points in [2,6] test: 200 points in [-6,6]

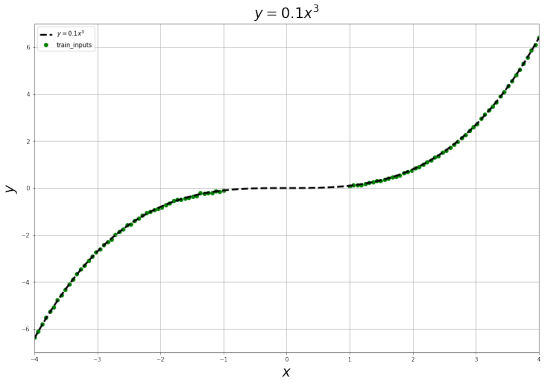
Table 4.2: Specifications of 1D datasets

A set of three different one-dimensional functions are used to evaluate the considered state-of-the-art

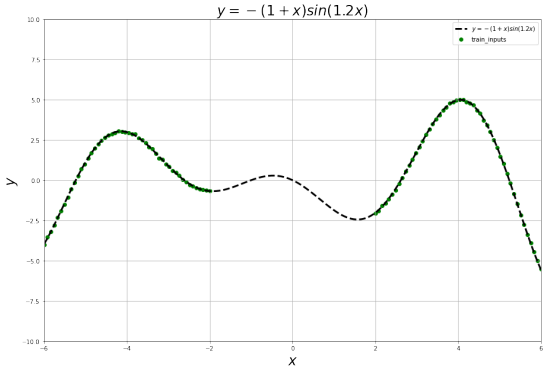
uncertainty methods. Use of low-dimensional datasets for evaluation has certain advantages: increased control over experimental parameters when compared to high-dimensional data, ease in visualizing results and reduced experimentation time. The following table lists the set of functions considered for this research work along with their specifications.



(a) $y = sin(3x)/3x + \epsilon, \epsilon \sim \mathcal{N}(0, (0.08)^2)$



(b) $y = 0.1x^3 + \epsilon, \epsilon \sim \mathcal{N}(0, (0.025)^2)$



(c) $y = -(1 + x)sin(1.2x) + \epsilon, \epsilon \sim \mathcal{N}(0, (0.04)^2)$

Figure 4.2: Functions in 1D dataset

Train and test data ranges for every function considered are chosen with an intent to evaluate the performance of uncertainty estimation methods in both within and outside the bounds of training data. For fn_1, the test data range is chosen to lie on either sides of the training data range. For the other two functions, the test data ranges are chosen to lie in between their train data ranges. A zero-mean Gaussian noise is added to training data of all three functions, with different values of standard-deviation. The set of functions are plotted in the Figure4.2.

4.2 Network Architectures

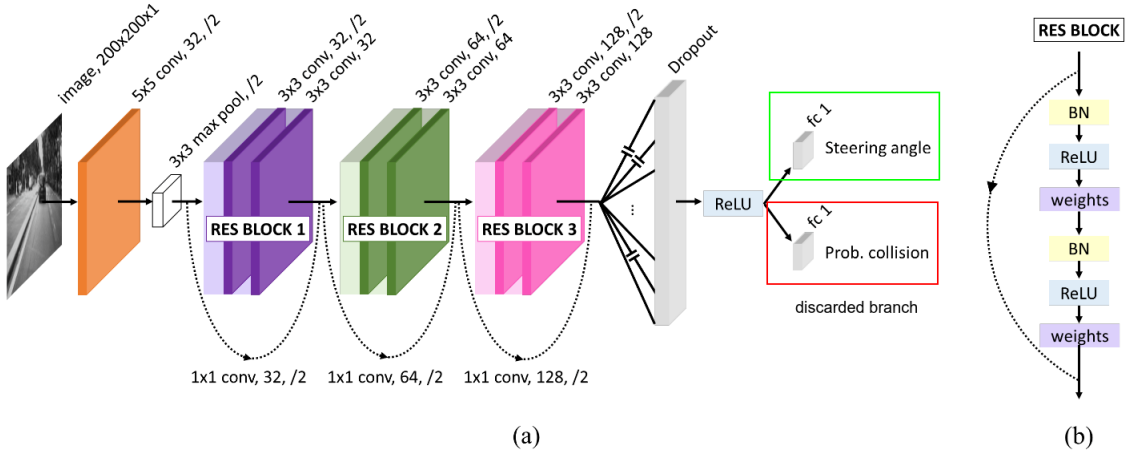


Figure 4.3: (a) Dronet architecture (b) Structure of every residual block . Collision classification output (bounded by the red box) is discarded and the steering angle prediction branch (bounded by the green box) is retained for this experiment. Image source: [25] (p.4)

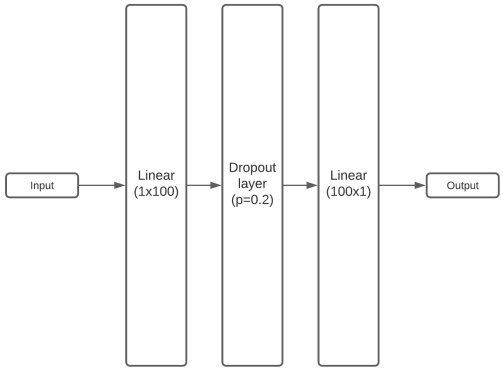


Figure 4.4: Block diagram of the neural network used with 1D datasets

4.2.1 Dronet

Dronet [25], a residual convolutional network architecture is used for experiments conducted on the steering angle dataset. The neural network is primarily designed to safely navigate a drone by performing the tasks of steering angle prediction (regression) and collision detection (binary classification). However, for the experiments conducted in this research work the collision detection output is not required and therefore its corresponding output branch in the neural network's output layer is discarded. Figure 4.3

depicts Dronet's architecture.

The model used for experiments takes a gray-scale input of size 200x200 and propagates it through a pair of convolution and max-pooling layers, three residual blocks, a dropout layer, a ReLU activation and finally a fully-connected (fc) layer which outputs predicted steering angles. When it comes to integrating MCDO_ADF (described in Section 3.7.2) with Dronet, new dropout layers are introduced before every convolutional layer at the test time. On the other hand, DER (described in Section 3.8) is integrated to Dronet by introducing three more output branches in the last fc layer for outputting parameters of the evidential distribution (described in Section 3.8.3).

4.2.2 Neural Network for 1D datasets

In order to conduct experiments on the set of 1D functions, a simple neural network architecture consisting of two fully-connected layers and a ReLU (Rectified Linear Unit) non-linearity is used (as shown in the Figure 4.4). The network is modified based on requirements of the integrated uncertainty estimation method. In the case of variant used with DER, the last fully connected layer is modified to output parameters of the evidential distribution. The MCDO_ADF variant of the neural network contains an additional dropout layer in order to facilitate extraction of MC samples, as described in 3.7. Also, after training the model is converted to its ADF equivalent.

4.2.3 Gaussian Process(GP) Models (1D Datasets)

A pair of GP models are included along side MCDO_ADF and DER for the evaluation on 1D datasets. GP can be understood as generalization of multi-variate normal distribution to infinite dimensions. In another perspective a GP represents the distribution over possible functions $f(x)$ that are consistent with given data, and is parameterized by mean $m(x)$ and covariance functions ($k(x, x')$). A GP can be represented as follows:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \tag{4.1}$$

GP models can be categorized into two types based on their assumptions about observation noise in data. Homoscedastic GPs assume independence between input data location and noise level while heteroscedastic GPs consider both the entities to be independent. In this research works both homoscedastic and heteroscedastic GPs are used for them to be compared with the two uncertainty estimation methods.

Rationale behind using GP Models for Benchmarking

"It has long been known that a single-layer fully-connected neural network with an i.i.d. prior over its parameters is equivalent to a GP, in the limit of infinite network width. This correspondence enables exact Bayesian inference for infinite width neural networks on regression tasks by means of evaluating the corresponding GP" [24].Existence of such a relationship between GPs and Bayesian Neural Networks

coupled with the ability of GPs to compute the predictive posterior distribution in a closed form, qualifies them to act as the baseline for evaluating the considered pair of uncertainty estimation methods.

4.3 Training details

4.3.1 Dronet

In order to benchmark the considered uncertainty estimation methods (MCDO_ADF and DER), a set of three Dronet models are used: 1. Vanilla version of Dronet 2. MCDO_ADF version of dronet with dropout layers after convolution layers 3. Evidential version of Dronet which uses the evidential loss function. Training details for those variants are provided in the upcoming sections.

Choice of certain hyperparameter values remains unchanged for training all the three models and are listed in the table below.

Hyperparameter	**Value**
Input image size (hxwxc)	200 x 200 x 1
Batch size	32
Training epochs	100
Learning rate	0.001
Dropout rate	0.2
Weight decay	0.0001
Learning rate decay	0.00001
Choice of optimizer	Adam
Initializers	Kaiming-normal for Conv2D in residual blocks, Xavier-uniform for Conv2D, linear layers in non-residual blocks, Constant initialization for batch normalization layers (weights with 1 and biases with 0)

Table 4.3: Hyperparameter specifications for training Dronet

Vanilla Dronet

A simple dronet model predicts steering angle for the given image input. Training the model involves reduction of the Mean Squared Error (MSE) loss, which is a popular choice for loss function in regression problems. Optimizing the MSE loss function intuitively means reduction of mean over euclidean distance (L2-Norm) between ground truth labels and predictions of observed data. The MSE loss function can be expressed as follows:

$$\mathbf{L}(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N}(y - \hat{y}_i)^2 \tag{4.2}$$

The technique of early stopping is used to avoid over-fitting, by saving model weights at the epoch corresponding to the least validation loss.

MCDO_ADF Dronet

This variant of Dronet is trained to facilitate estimation of uncertainty associated with its predictions using the MCDO_ADF technique. The training procedure for this variant remains unchanged from the Vanilla variant except for the introduction of dropout before every convolution layer. Though it is sufficient to have the vanilla variant for applying this method, dropout was used during training with an intent to regularize the process. The inference procedure for MCDO_ADF applied models is clearly explained in Section 3.7.3. There are three hyper-parameters additionally required for the procedure: 1. Monte-Carlo (MC) sample count 2. Noise-variance 3. Minimum-variance. The value of MC sample count has a direct impact on the inference time as it determines the number of forward passes for a given input to determine model uncertainty. For this experiment, we set its value to be 20 to replicate results provided in [26]. Both the values of noise and minimum variances are chosen to be 0.001. Noise variance indicates the level of sensor noise and minimum-variance signifies the minimum value of noise-variance to be propagated through every layer.

Evidential Dronet

The evidential Dronet model outputs parameters of the evidential distribution (described in Section 3.8.3) for a given input image. The distribution parameters can be in turn used to compute the mean (predictive mean) and uncertainty associated with it. Except for the choice of evidential loss function (described in Section 3.8.4) for this model , training criteria remains unchanged from the vanilla variant.

4.3.2 Neural Network with 1D Dataset

Similar to the case of Dronet, two variants of the simple network described in Section 4.2.2 are used for evaluation. The MCDO_ADF variant is trained using MSE loss, while evidential loss is used for its evidential counterpart. Also, the MCDO_ADF variant uses dropout during test time. The following table contains values of hyperparamters that remain common to both the variants.

Hyperparameter	**Value**
Batch size	Training data set size
Optimizer	Adam
Learning rate	0.001
Epochs	20000
Initializers	Normal initializer for weights and zeros for bias

Table 4.4: Choice of hyperparameters for neural nets used with 1D datasets

4.3.3 Gaussian Process Models

Parameter	**Homoscedastic GP**	**Heteroscedastic GP**
Covariance function kernel	Matern kernel	Two Squared-exponential kernels as independent priors for likelihood mean and observation noise
Likelihood function	Gaussian	Heteroscedastic likelihood with a Normal conditional distribution
Length scale initial value	0.3	1.0
Optimizer	Scipy (L-BFGS-B based optimizer in gpflow)	Gradient descent + Adam optimizers
Maximum iterations	100	100
Inducing points for variational inference	Not Applicable	20

Table 4.5: Training details of GP models

5

Experimental Evaluation

5.1 Metrics

5.1.1 Root Mean Squared Error (RMSE)

Root Mean Squared Error (RMSE), measures the spread of distances between model predictions and their corresponding ground truth values. Alternatively, it can be explained as the standard deviation of prediction errors. RMSE is a well-known accuracy metric in regression problems. The metric is non-negative in nature, with lower values indicating better model fit.

$$\textbf{RMSE} = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}} \tag{5.1}$$

Here y,$\hat{y}$, and N represent ground truth labels, predictions and number of data points respectively.

5.1.2 Explained Variance (EVA)

The Explained Variance (EVA) is a measure of a regressor's ability to capture variance(variation) of given data. The metric can be computed with the following expression:

$$\textbf{EVA} = 1 - \frac{Variance(y - \hat{y})}{Variance(y)} \tag{5.2}$$

Here y, $\hat{y}$ represent ground-truth labels and predictions respectively. The numerator term denotes variance of residuals whereas the denominator denotes the underlying variance in ground-truth labels. For an ideal regressor, the value of EVA equals 1.

5.1.3 Negative-Log-Likelihood (NLL)

In this research work, the NLL metric is used compare performances of the uncertainty estimation methods. NLL for a given pair of prediction and uncertainty can be computed as follows

- A distribution (often Gaussian) is created with the model prediction as its mean and uncertainty as its variance.

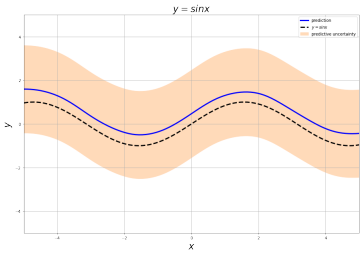
- The conditional probability of observing the ground-truth label (corresponding to the input) in the created distribution is determined. This is nothing but the likelihood value of ground-truth in the created distribution.

- In order to handle and effectively represent very low values of likelihood, negative of natural logarithm is applied to the value. After application of negative logarithm, the total likelihood can be computed by summing all individual values.

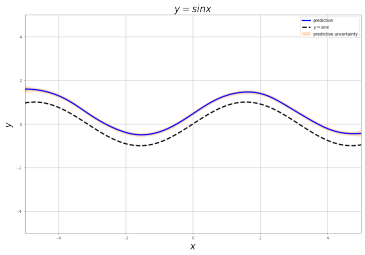
$$\mathbf{NLL} = -\sum_{i=1}^{N} \ln P(y_i|\mathcal{N}(\hat{y}_i, \sigma^2)) \tag{5.3}$$

Here y,$\hat{y}$,σ^2 and N represent ground truth labels, predictions, predictive uncertainty and number of data points respectively. Lower the value of NLL better the performance of an uncertainty estimation method associated with it. However, the value of NLL highly depends on the number and choice of test points used for evaluation. Therefore, the metric can be used to compare performance of uncertainty estimation methods only when the same data set is used for their evaluation.

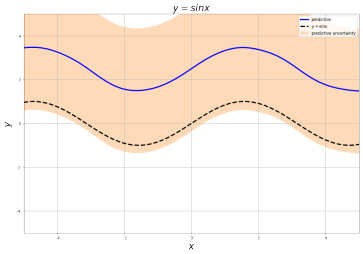
Limitations of NLL



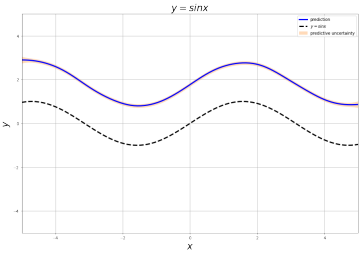
(a) Close prediction with high uncertainty



(b) Close prediction with low uncertainty



(c) Far prediction with high uncertainty



(d) Far prediction with low uncertainty

Figure 5.1: Plots to explain effects of predictions and confidence bounds on NLL

Case	NLL	RMSE	EVA
Close prediction with high uncertainty	1.04	0.50	0.99
Close prediction with low uncertainty	87.46	0.50	0.99
Far prediction with high uncertainty	4.05	2.50	0.99
Far prediction with low uncertainty	2220.67	2.50	0.99

Table 5.1: Different cases to describe the impact of prediction distance and uncertainty levels on metrics

NLL measures the goodness of fit of the approximated posterior distribution to the true function's mean(ground truth). The metric however fails to evaluate fidelity of the approximated posterior distribution. Therefore, NLL "may be a good criteria for model selection, it is not a reliable criteria for determining how well an approximate posterior aligns with the true posterior" [40].

Plots in the Figure 5.1 depict different possible cases of predictions and uncertainty levels whose corresponding NLL, RMSE and EVA values are tabulated in Table 5.1. It can be inferred from Figure 5.1a and Figure 5.1b that irrespective of both predictions being identical, width of confidence intervals has a huge impact on their NLL values. Likewise, the prediction in Figure 5.1b is much closer to the target than the one in Figure 5.1c, yet the former is penalized heavily in terms of NLL (2220.67) due to its low value of uncertainty. However, values of EVA remain constant for all scenarios and RMSE changes only when there is a shift in the prediction. This signifies that the interpretation of NLL as a metric for uncertainty estimation quality holds good only when it is considered along with a measure of predictive accuracy like RMSE. Also, the metric can be used to compare methods only when they are evaluated on a given dataset.

5.2 Predictive Accuracy and Uncertainty Quality

RQ2. How do the identified uncertainty estimation methods compare with each other and Gaussian Process models, based on their uncertainty estimation quality and prediction accuracy of their respective models?

5.2.1 Udacity Steering Angle Dataset

Quantitative Comparison

Model	RMSE	EVA	NLL
Vanilla Dronet	0.034	0.98	NA
MCDO_ADF Dronet	0.151	0.68	-0.74
Evidential Dronet (squared version of loss)	0.022	**0.99**	**-0.94**
Evidential Dronet (L1 norm version of loss)	**0.021**	**0.99**	0.31

Table 5.2: A quantitative comparison of uncertainty estimation methods when applied to Dronet

- It can be inferred that Evidential Dronet outperforms the other two models in terms of both predictive accuracy and quality of uncertainty estimation (NLL).

- In the case of predictive accuracy expressed in terms of RMSE, Evidential Dronet outperforms the Vanilla variant only by a slight margin. However, difference in RMSE between Evidential and MCDO_ADF variants is considerable. This could be partly attributed to the fact that both predictions and uncertainty estimates outputted by the MCDO_ADF variant depend on the count of Monte-Carlo (MC) samples considered. Higher the value of MC samples better the model's performance (refer Figure 5.2). However, this holds true only until a particular value of MC sample count. For this experiment, 20 MC samples are considered.

- The relationship between MC sample count and predictive accuracy holds good for the Explained Variance (EVA) measure as well.

- The quality of predictive uncertainty estimated by the Evidential Dronet expressed in terms of NLL is better than the MCDO_ADF variant. This intuitively means that the former technique is able to determine parameters of distributions in which likelihood of finding ground truth labels is higher than in distributions outputted by latter.

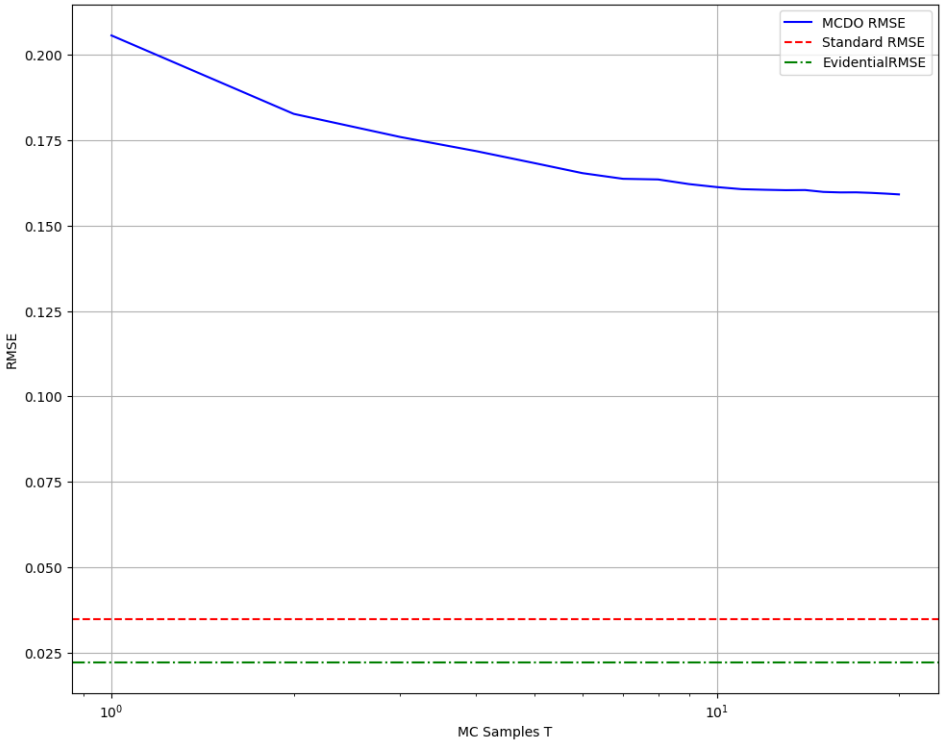


Figure 5.2: Plot depicting relationship between the MC sample count and RMSE during the MCDO_ADF model inference

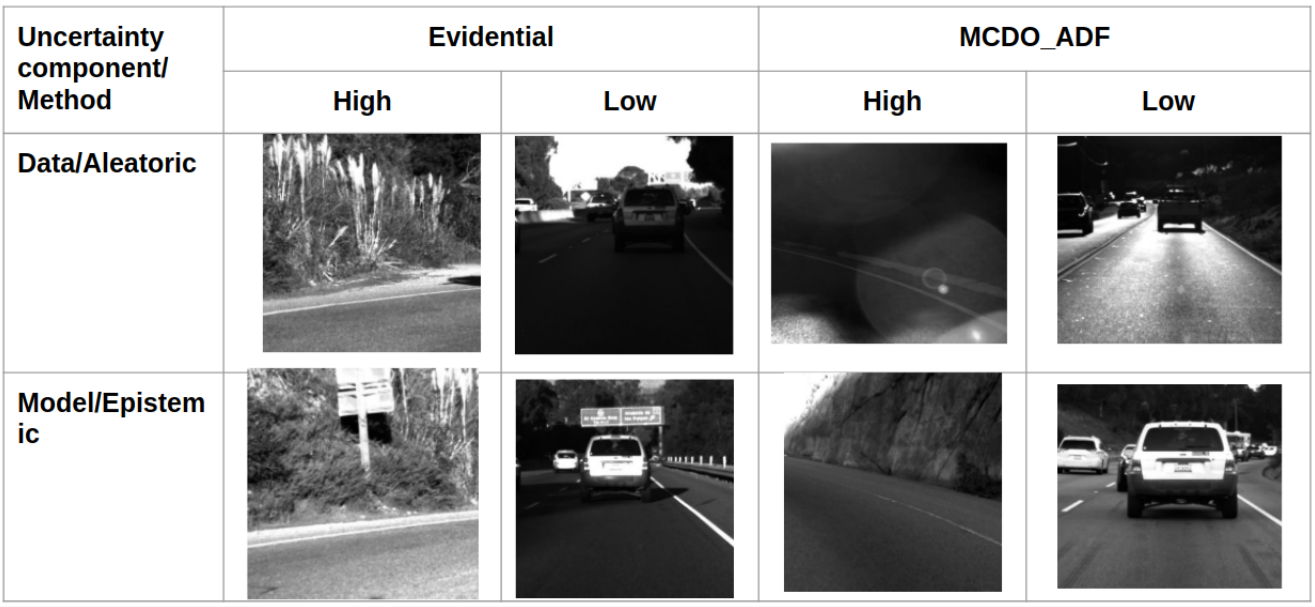


Figure 5.3: A set of sample images with extreme levels of uncertainties predicted by both the methods

Qualitative Comparison

Category	Method	Analysis
Aleatoric	MCDO_ADF	• The method performs well in estimating data uncertainty.The presence of glare, blur, and poor illumination in images reported with high values show that the estimated uncertainty is indeed aleatoric. • Most of the images reported with low values of aleatoric uncertainty are characterized by high contrast and low noise levels. • There also exist certain over-illuminated images for which the method reports a low value of data-uncertainty and is undesirable.
Aleatoric	DER/Evidential	• The method reports a high-value of aleatoric uncertainty for blurry and unclear images and is desirable. • It is interesting to note that this method reports high-values of data uncertainty for images with objects such as grass that produce patterns similar to noise. • Similar to MCDO_ADF this method performs well in reporting low aleatoric uncertainty for clear images except for the ones with shadows.
Epistemic	MCDO_ADF	• The method reports high values of epistemic uncertainty for both images with unclear lane markings (intuitively a key feature to predict steering angles) and noise induced by factors such as blur and glare. • Reporting high values of epistemic uncertainty for noisy images proves the method's ability to jointly model both the components of uncertainty. • Similar to the aleatoric case, the method reports low values of epistemic uncertainty for clear images except for the ones with poor and over illumination.
Epistemic	DER/Evidential	• The method produces high values of model uncertainty for both images with unclear features and noise, similar to MCDO_ADF. • However, there is a difference between images reported with high values of epistemic uncertainty between MCDO_ADF and evidential showing the fact that they both have different criteria for uncertainty estimation. • The set of images reported with low values of epistemic uncertainty have a lot of similarities with that of ones with low-aleatoric variances.
Total	MCDO_ADF and DER/Evidential	The set of images reported with high/low values of predictive uncertainties by both the methods correspond to the ones with high/low values of their components (aleatoric and epistemic)

Table 5.3: A qualitative comparison of uncertainty estimation methods

5.2.2 1D Dataset

Function y=f(x)	**Metric**	**Homoscedastic GP**	**Heteroscedastic GP**	**Evidential**	**MCDO_ADF**
$y = \frac{sin(3x)}{3x}$	NLL	**-0.82**	-0.51	*0.57*	2.27
	RMSE	**0.07**	0.08	0.40	*0.39*
	EVA	**0.94**	0.91	*-0.29*	-0.43
$y = 0.1x^3$	NLL	**-2.44**	-1.77	*-0.15*	1.11
	RMSE	**0.01**	0.03	0.23	*0.11*
	EVA	**0.99**	**0.99**	**0.99**	**0.99**
$y = -(1+x)sin(1.2x)$	NLL	**-1.02**	1.91	*6.14*	>>1(182049)
	RMSE	0.61	**0.15**	2.68	*0.8*
	EVA	0.94	**0.99**	*-0.05*	0.87

Table 5.4: A quantitative comparison of uncertainty estimation methods on 1D datasets

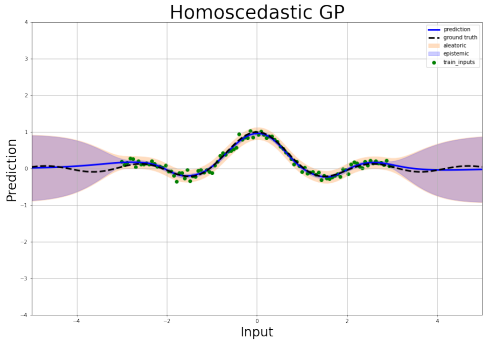
- It can be inferred from the tabulation that GP outperform the considered pair of uncertainty estimation methods in terms of every metric (values marked in bold). However, the pair GP models are included in this comparison only as a baseline.
- When it comes to comparing DER with MCDO (italicized values correspond to better performance), the former performs better in terms of NLL and EVA while the latter has relatively lower values of RMSE, the measure of predictive accuracy.
- Better performance of DER over MCDO_ADF can be attributed to three important factors:
  - Flexibility: The extent of confidence interval around a prediction estimated by a method.
  - Quality of model fit to data
  - Performance in OOD regions (discussed more in the next section).
- MCDO_ADF suffers from high values of NLL (undesirable) in case of all the three functions due to lack of flexibility.

Qualitative Comparison

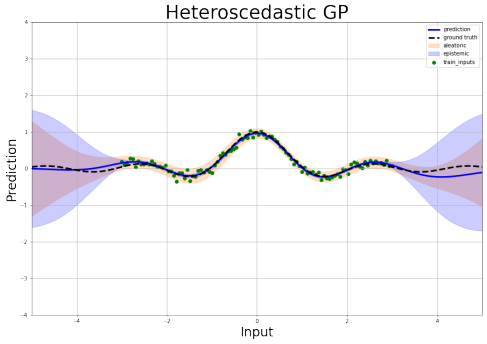
- As it can be witnessed from Figure 5.4d, Figure 5.7b and Figure 5.4l DER provides a constant estimate for the value of aleatoric uncertainty similar to homoscedastic GP, which is desirable.
- The MCDO_ADF technique performs relatively better than DER in estimating epistemic uncertainties for fn_1 (refer Figure 5.4c), while its performance plummets in the case of fn_3 (refer Figure 5.7f) where it fails to report high values of uncertainty for the OOD region which results in a very high value of NLL.

- The MCDO_ADF variant of the neural network model performs better than its DER counterpart in modeling smoothness of functions. While in the case of GPs, the use of squared-exponential kernels as priors leads to better modeling of such functions.

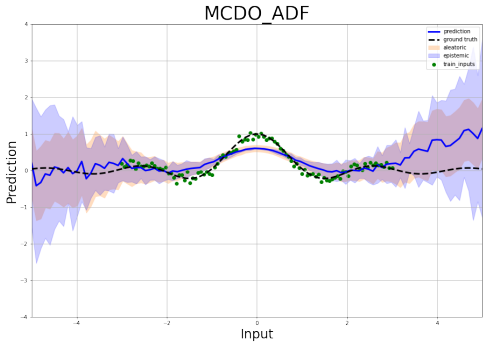
- The success of homoscedastic GPs in modeling aleatoric uncertainties can be attributed to constancy in variance of Gaussian distribution used for adding noise to input data generated from a given function.



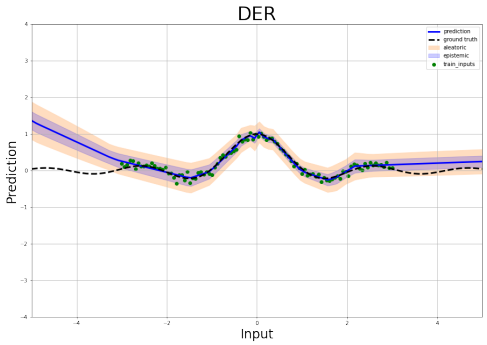
(a) Homoscedastic GP on fn_1: $y = \frac{sin(3x)}{3x}$



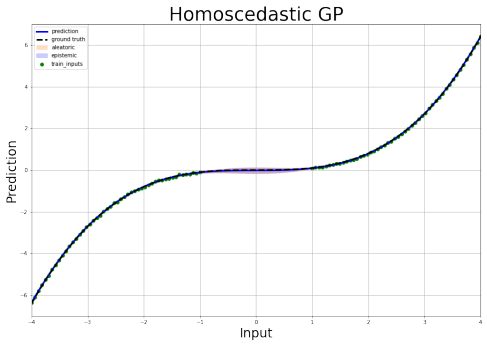
(b) Heteroscedastic GP on fn_1: $y = \frac{sin(3x)}{3x}$



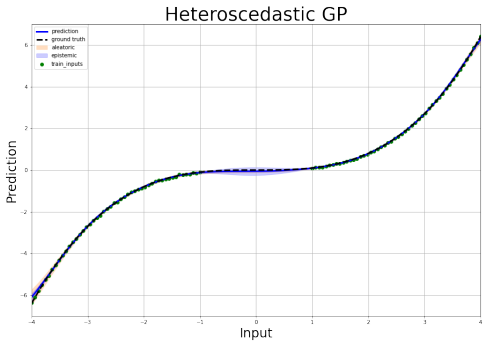
(c) MCDO_ADF on fn_1: $y = \frac{sin(3x)}{3x}$



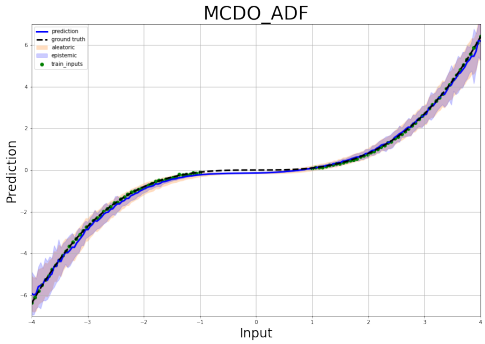
(d) DER on fn_1: $y = \frac{sin(3x)}{3x}$



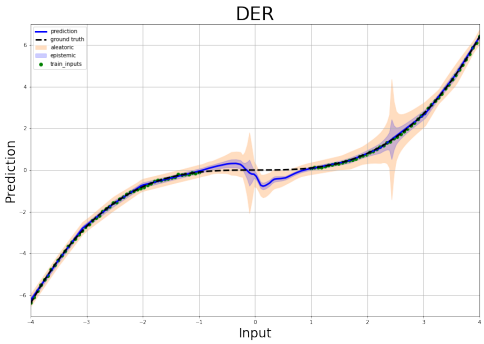
(e) Homoscedastic GP on fn_2: $y = 0.1x^3$



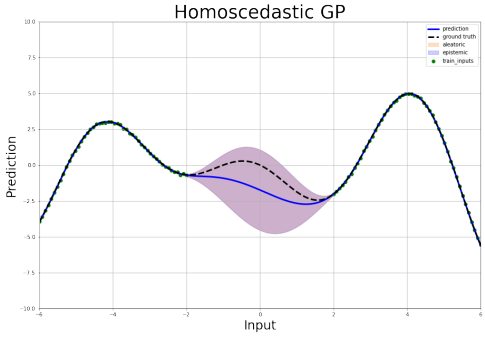
(f) Heteroscedastic GP on fn_2: $y = 0.1x^3$



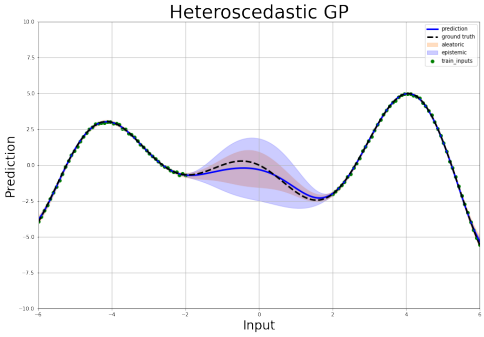
(g) MCDO_ADF on fn_2: $y = 0.1x^3$



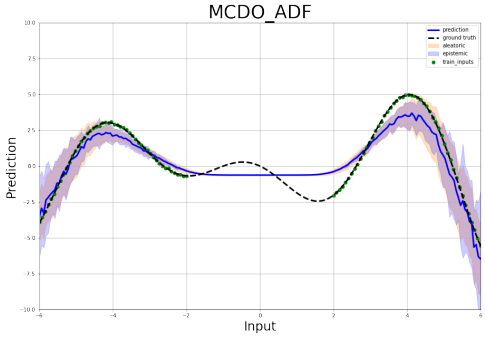
(h) DER on fn_2: $y = 0.1x^3$



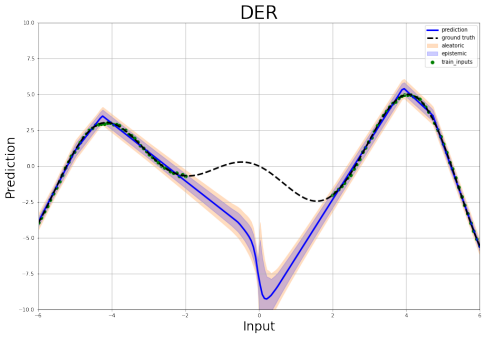
(i) Homoscedastic GP on fn_3: $y = -(1 + x)sin(1.2x)$



(j) Heteroscedastic GP on fn_3: $y = -(1 + x)sin(1.2x)$



(k) MCDO_ADF on fn_3: $y = -(1 + x)sin(1.2x)$



(l) DER on fn_3: $y = -(1 + x)sin(1.2x)$

Figure 5.4: Plots depicting application of uncertainty estimation methods on 1D datasets

5.3 Out-Of-Distribution (OOD) Testing

RQ3. How do the identified uncertainty estimation methods compare with each other based on their response to OOD and adversarially perturbed inputs?

One of the important uses for having an estimate of uncertainty associated with a neural net model's output is "selective prediction". This means that the confidence estimate can be used to determine the correctness of an output and in turn decide whether to consider it for further processing/decision-making or not. It is crucial for an uncertainty estimation method to produce a higher value of uncertainty when its model faces test samples from an unknown data distribution. This section evaluates the considered uncertainty estimation methods on their response to out-of-distribution inputs.

5.3.1 Response to OOD Data

Steering Angle Dataset

In the qualitative comparison (Section 5.2.2) of methods on the steering angle data set, the relationship between uncertainty estimates and factors such as presence of blur, glare, sun-flare and illumination was discussed. In this section, the speculated relationship is validated. Using image processing techniques, three sets of images are generated by introducing fog, snow-like patterns, and darkness to a noise-less image (Figure 5.5) chosen from the steering angle dataset. These images are fed as inputs to both MCDO_ADF and DER model variants and responses of their respective uncertainty estimation methods are evaluated.

Analysis

Please refer plots in Figure 5.7.

- Darkness: Increasing levels of darkness almost does not have any impact on total uncertainty estimated by DER. While an increase in uncertainty estimates of MCDO_ADF can be observed for initial levels of darkness, the trend do not remain for higher values which makes the method's response undesirable. When it comes to predictive error, absolute deviation values of the DER model variant is better aligned with increasing levels of darkness than its MCDO_ADF counterpart.
- Fog: Though there exist few inconsistencies, MCDO_ADF's estimates of total uncertainty are better aligned with increasing levels of fog than DER. The same holds true for the relationship between fog levels and predictive error as well.
- Snow: While DER's response in terms of uncertainty estimates remains almost constant to changes in levels of snow, a peculiar trend can be observed in MCDO_ADF's output that the values of uncertainty decrease with increase in snow levels. There does not exist a strong correlation between the levels of snow and predictive error values corresponding to both the model variants.

Except for the case of images with different levels of fog where MCDO_ADF shows a desirable response, any significant relationship between the considered set of variables could not be established, in the other

two cases. Observations such as unchanging values of uncertainty estimated by DER for increasing levels of fog and snow, and existence of alignment between uncertainty estimates and predictive error of the DER model variant in the case of dark images, show that factors such as model calibration and invariance of a model to a feature have a role to play. In order to better understand responses of the chosen pair of uncertainty estimation methods to OOD inputs, the analysis is extended to the test split of the steering angle dataset.

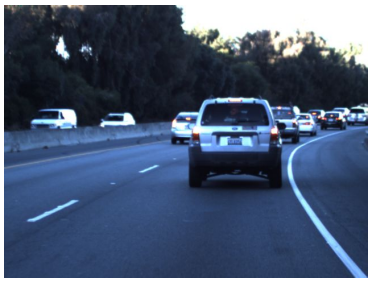


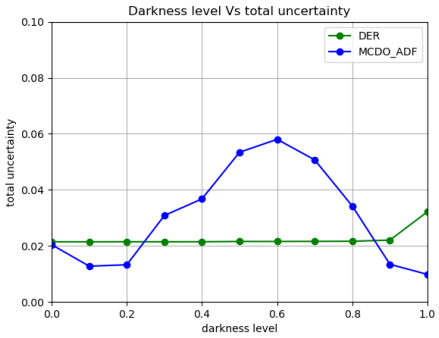
Figure 5.5: Image considered for OOD analysis

(a) darkness_coeff = 0.1 (b) darkness_coeff = 0.3 (c) darkness_coeff = 0.5 (d) darkness_coeff = 0.7 (e) darkness_coeff = 0.9

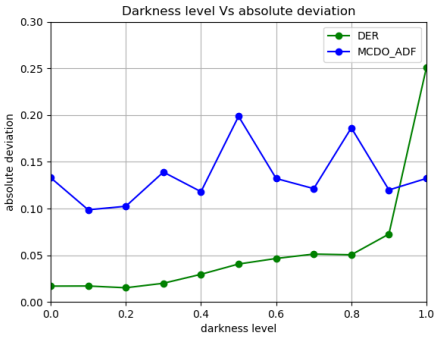
(f) fog_coeff = 0.1 (g) fog_coeff = 0.3 (h) fog_coeff = 0.5 (i) fog_coeff = 0.7 (j) fog_coeff = 0.9

(k) snow_coeff = 0.1 (l) snow_coeff = 0.3 (m) snow_coeff = 0.5 (n) snow_coeff = 0.7 (o) snow_coeff = 0.9

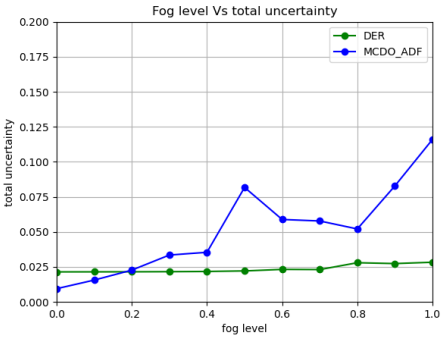
Figure 5.6: A subset of synthesized images introducing increasing levels of darkness, fog and snow



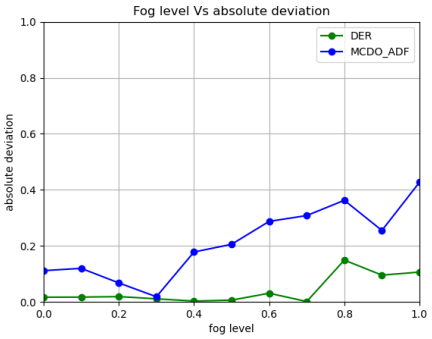
(a) Levels of darkness Vs Total uncertainty



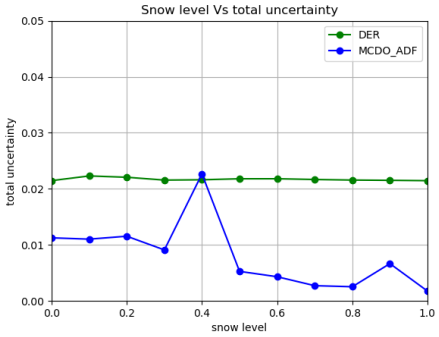
(b) Levels of darkness Vs Absolute deviation



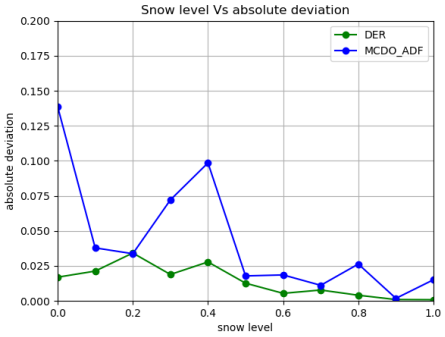
(c) Levels of fog Vs Total uncertainty



(d) Levels of fog Vs Absolute deviation



(e) Levels of snow Vs Total uncertainty



(f) Levels of snow Vs Absolute deviation

Figure 5.7: Response to OOD images (steering angle dataset)

Extended Analysis

Plots in Figure 5.8 depict the change in values of uncertainties (averaged over all outputs) and RMSE with respect to the increase in darkness levels of input images. Existence of an almost linear relationship between increasing darkness level in images and prediction error (RMSE) values of both DER and MCDO_ADF models is observed. This indicates that both the models are well-calibrated. The MCDO_ADF method

clearly outperforms DER in appropriateness of its response to increasing darkness levels in images, by producing high-valued uncertainty estimates. However its response is inconsistent as uncertainty values (epistemic, aleatoric and total) surge sharply as the darkness coefficient value exceeds 0.7. The reason for this sudden drop could not be determined. In the case of DER, uncertainty levels mostly remain constant and few fluctuations occur. Therefore, any significant relationship could not be established between darkness levels and corresponding uncertainty estimates produced by both the methods.

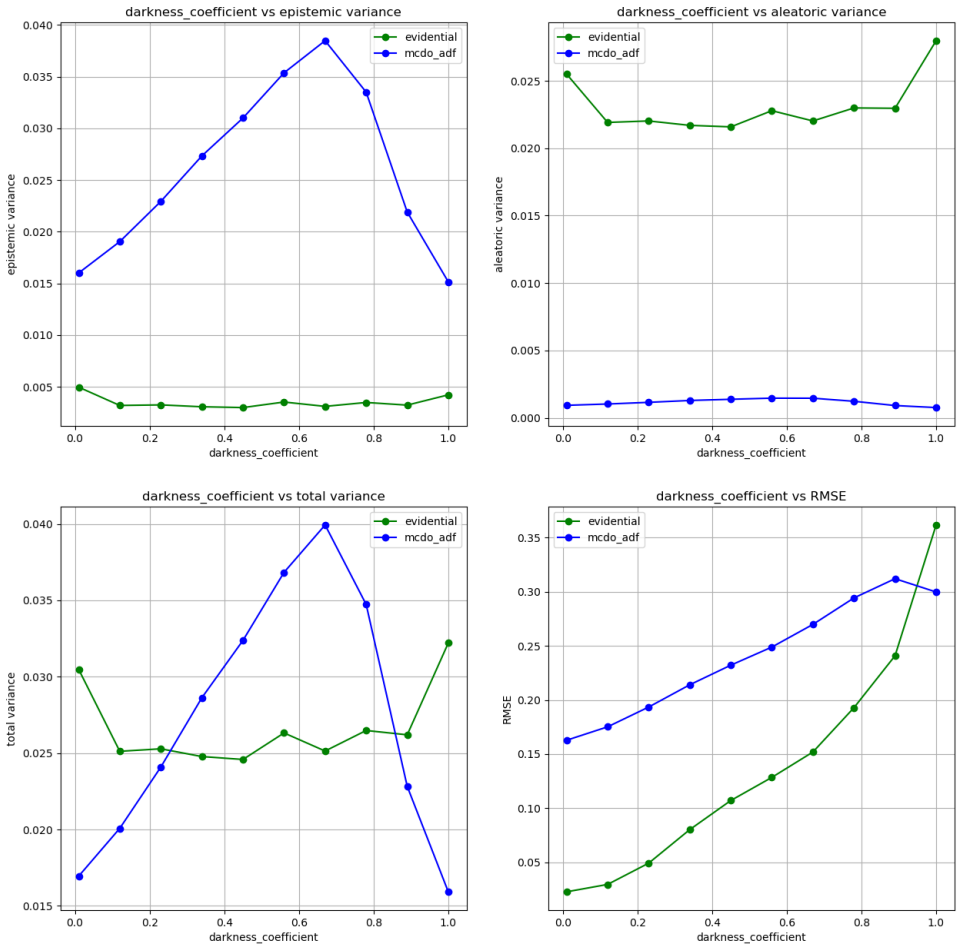


Figure 5.8: Response to OOD images (extended analysis)

1D Datasets

Train and test data ranges of the set of 1D functions (refer Figure 5.4) are set in such a way that the OOD regions lie both in-between and on either sides of training data ranges. Owing to the smooth nature of target functions and use of squared exponential kernels GPs perform well both in terms of predictive accuracy and NLL (uncertainty estimation quality) in OOD areas. Both the considered uncertainty estimation methods do not perform well in predicting the mean and estimating uncertainties in OOD regions of fn_3. Due to narrow confidence intervals proposed by MCDO_ADF in this region, the method performs poorly in terms of NLL. In the case of fn_2, predictions of both the methods remain close to the target. The MCDO_ADF method outperforms DER by reporting a high value of epistemic uncertainty for OOD regions in fn_1.

5.3.2 Response to Adversarial Attacks (for Steering Angle Dataset only)

(a) $\epsilon = 0$ (b) $\epsilon = 0.02$ (c) $\epsilon = 0.06$ (d) $\epsilon = 0.08$ (e) $\epsilon = 0.10$

Figure 5.9: Increasing levels of adversarial noise

"Adversarial examples are malicious inputs designed to fool machine learning models" [20]. Such data samples can be considered as an extreme case of OOD as they are synthesized by perturbing inputs in an adversarial fashion to cause maximum error on the model prediction. An uncertainty estimation method needs to be capable of identifying and reporting an adversarially perturbed input data sample by producing a high value of uncertainty. In order to evaluate responses of the considered uncertainty estimation methods, samples from training set distribution are perturbed using the FGSM (Fast Gradient Sign Method) [14] and fed to models. A set of images produced from a given image subject to different levels of adversarial perturbations is shown in the Figure 5.9. Adversarial images fed to a given Dronet model variant are synthesized using perturbations generated by its own. Plots in the Figure 5.10 depict the impact of increasing perturbation levels (denoted by ϵ) on average values of uncertainty estimates and RMSE values respectively, for both model variants. The following can be inferred from the set of plots in the Figure 5.10:

- The existence of linear relationships between ϵ , RMSE and total variance respectively is desirable and indicates that the MCDO_ADF model variant is better calibrated (alignment between error and uncertainty) than its DER counterpart.

- Though aleatoric uncertainties estimated by DER are more sensitive to increasing levels of ϵ than MCDO_ADF, the method's response is non-linear in nature.

- Epistemic uncertainty forms the major component of total uncertainty estimated by MCDO_ADF while the aleatoric component dominates in the case of DER.

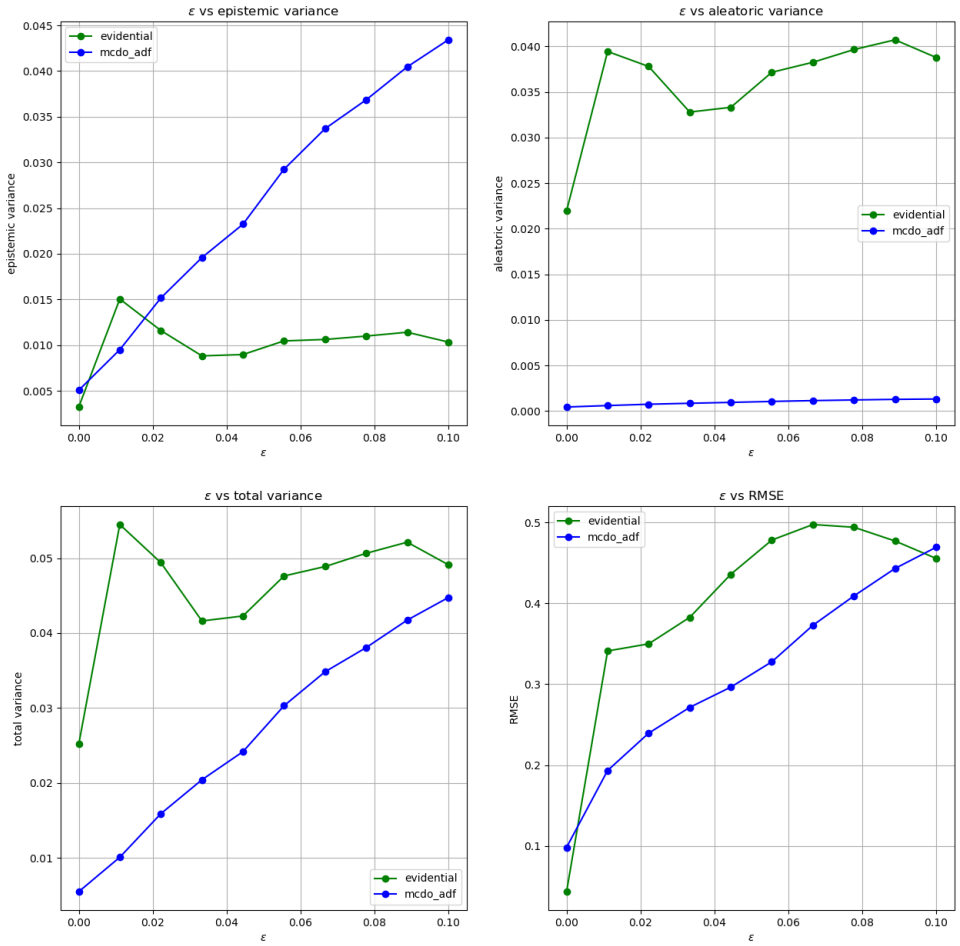


Figure 5.10: Plots of ϵ (adversarial perturbations) against uncertainties and RMSE

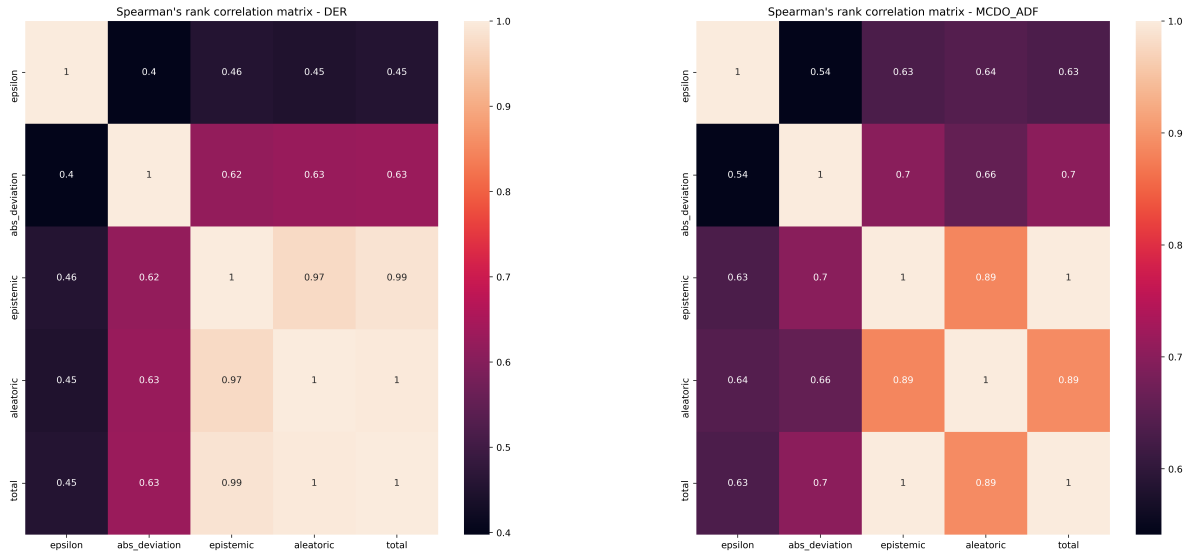
Spearman's Correlation Analysis

Spearman's rank correlation (often denoted by ρ) is a measure of statistical dependence between a pair of variables. Intuitively ρ indicates the strength and direction of association between two ranked variables. ρ varies between -1 and 1. The direction of association is indicated by sign of ρ while its magnitude indicates the strength of correlation between the variable pair.

In the context of analyzing response of uncertainty estimation methods to adversarial attacks, Spearman's rank correlation coefficient is computed for pairs of following variables : ϵ, absolute deviation of a prediction from its corresponding ground truth value, aleatoric, epistemic and total uncertainties. Though Pearson's correlation coefficient is a common choice for correlation analysis, there are two reasons to prefer Spearman's correlation for this analysis:

- Pearson's correlation measure applies to normally distributed variables. As no assumptions are made on underlying distributions of the considered set of variables Spearman's correlation coefficient is preferred.
- Spearman's coefficient measures correlation between a pair of ranked variables. As this analysis focuses on effects of increasing levels of ϵ (a ranked variable) on other variables, the measure becomes an apt choice.

Spearman's correlation coefficients are arranged in form of a symmetric matrix (indices denoting variables) separately for both MCDO_ADF and DER model variants as shown in the figure below.



(a) Spearman's correlation matrix for DER

(b) Spearman's correlation matrix for MCDO_ADF

Figure 5.11: Spearman's correlation heatmaps

The following can be inferred from the computed correlation coefficient matrices:

- MCDO_ADF shows relatively a stronger correlation between ϵ and other variables of interest than DER. This indicates a higher-level of alignment between ϵ and uncertainties estimated by MCDO_ADF, which is desirable.
- A stronger association exists between epistemic and aleatoric components of uncertainty estimated by DER ($\rho = 0.97$) than MCDO_ADF ($\rho = 0.89$). This can be attributed to DER's ability to relate the pair of uncertainty components.
- The dominance of aleatoric and epistemic uncertainty components in total uncertainties estimated by DER and MCDO_ADF respectively, can be observed in their corresponding correlation coefficients. This aligns with inferences obtained from Figure 5.10.

5.4 Evaluation Summary

Experiment	**Metric**	**Better performer**
Udacity steering angle test dataset evaluation	NLL	DER
	RMSE	DER
	EVA	DER
	Qualitative comparison	MCDO_ADF
1D datasets	NLL	DER
	RMSE	MCDO_ADF
Response to adversarial attacks	Spearman's rank correlation coefficient between perturbations and uncertainty	MCDO_ADF
Response to OOD inputs (1D datasets)	NLL	DER
	RMSE	MCDO_ADF

Table 5.5: Table of conducted experiments and their results

The above table summarizes results from the set of experiments conducted in this research work. It can be inferred that there does not exist a single clear winner which outperforms the other in every conducted experiment. However, when it comes to evaluating the pair of methods based on uncertainty estimation quality, measured in terms of NLL, DER outperforms MCDO_ADF in both the datasets. This signifies the higher likelihood of ground truth's presence in distributions outputted by DER integrated models, than its MCDO_ADF counterparts. MCDO_ADF performs better in terms of RMSE, a measure of predictive accuracy on both datasets. This can be attributed to the use of multiple MC samples by MCDO_ADF models to output their predictions. When it comes to analyzing the pair of methods on OOD inputs, MCDO_ADF's response is desirable when inputs are adversarially perturbed, while DER's uncertainty estimates are more reliable when a model encounters inputs that lie outside the training data distribution.

6

Conclusions

This research work focuses on benchmarking state-of-the-art uncertainty estimation methods in neural networks meant for the task of regression. The intent of this work is to evaluate modern uncertainty estimation methods on various aspects, in order to aid Deep Learning (DL) practitioners choose a suitable method based on their requirements, and also to let the DL research community know potential methods to be considered for enhancements and further research. A survey of methods is conducted to choose the top five state-of-the art methods for further analysis. Based on claims made by authors and deficits analysis, two (MCDO_ADF and DER) out of five methods are chosen for an intensive experimental evaluation. The chosen pair of methods are evaluated using the openly available Udacity steering angle dataset that corresponds to a safety-critical application and a set of three datasets generated from 1D functions. The selected uncertainty estimation methods are evaluated on: the quality of estimated uncertainty, their impact on predictive accuracies of host models and responses to OOD data inputs. DER integrated models output more accurate predictions and uncertainty estimates of higher quality than the other, while MCDO_ADF performs desirably in responding to OOD inputs.

6.1 Contributions

Following is the list of important contributions made by this work:

- Literature review and identification of state-of-the-art uncertainty estimation methods for regression nets.
- Extensive description and comparison of MCDO_ADF and DER approaches, on Udacity steering angle dataset and three synthesized 1D datasets.
- Implementation of GP models and comparison with MCDO_ADF and DER on 1D datasets.
- Identification of metrics and detailed analysis of NLL as a measure of uncertainty estimation quality.
- Evaluation of responses of uncertainty estimation methods to adversarially perturbed and OOD inputs and its corresponding implementation.
- Spearman's rank correlation analysis of uncertainty estimation methods on adversarially perturbed inputs from Udacity steering angle dataset.

6.2 Lessons Learned

The following insights were gained during the course of this research work:

- It is preferred for an uncertainty estimation method to learn how to model uncertainties from training data rather than estimating uncertainty during test-time with some implicit assumptions.
- A hierarchy of distributions can be used to model both the likelihood distribution and distributions over its parameters simultaneously.
- There does not exist a metric to evaluate fidelity of uncertainty estimation methods, in approximating the exact posterior predictive distribution, especially in the regression setup. The NLL metric only measures the goodness of fit of ground truth in the predicted distribution.
- It is crucial to evaluate uncertainty estimation methods not only based on metrics but also based on their role in the end application such as selective prediction.
- It is important for any uncertainty estimation method to consider and model the relationship between epistemic and aleatoric uncertainty components.

6.3 Future Work

DER has some clear advantages over MCDO_ADF in aspects such as reduced inference time and learning how to model uncertainties from training data. The former can be further improved by:

- Experimenting with different choices for the prior distribution over likelihood, such as Normal-Wishart or Log-Normal distributions. A recently published work "Regression Prior Networks" [28] focuses on a similar approach.
- Using better strategies to distinctively optimize Data-fit and Regularizer terms of the evidential loss function.
- Understanding the notion of evidence (belief mass) in relation to the application at hand, for enhancing interpretability of outputted uncertainty estimates.

In general, there is a need for devising a metric that can evaluate uncertainty estimation methods in the regression setup, based on their ability to approximate the predictive posterior distribution, unlike NLL. A success in the extension of GP to be used with high-dimensional data such as images, can be a significant breakthrough in addressing the problem of uncertainty estimation.

A

Metrics

An extensive review was conducted to find a suitable metric that overcomes limitations of Negative-Log-Likelihood (NLL) in capturing the quality of approximated posterior distribution. The following metrics were considered for analysis:

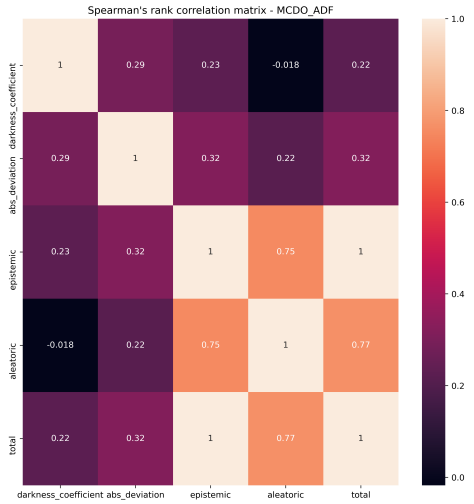
- Brier score
- Expected Calibration Error
- Maximum Calibration Error
- Bayesian Information Criterion
- Mutual Information
- Entropy
- Cross Entropy
- Kullback-Leibler Divergence
- Signal to Noise Ratio

Though some of them suit for measuring calibration, sharpness and uncertainty estimation quality in a classification setup, none of them apply to regression due to the continuous and unbounded nature of its output space. Understanding and approaching Deep Neural Networks from the perspective of Information theory [32] may prove helpful in devising a suitable metric to measure the quality of uncertainty estimates in the regression setup.

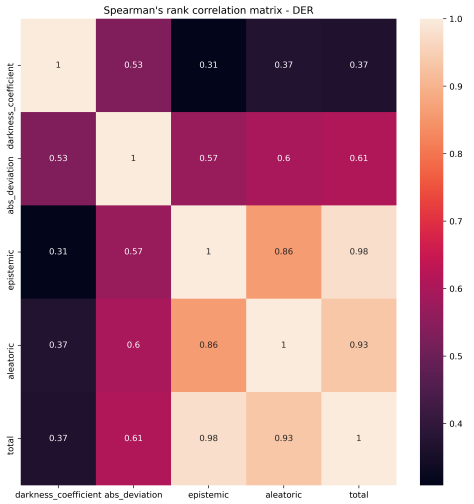
B

Correlation matrices for OOD Analysis

The following are Spearman's correlation coefficient matrices for MCDO_ADF and DER methods, obtained by relating variables involved in the extended OOD response analysis described in 5.3.1. Apart from the strong correlation value between darkness levels (darkness_coefficient) and prediction error (abs_deviation), values of Spearman's coefficient for other pairs of variables do not mean much as they do not have a monotonic relationship (refer Figure 5.8), which is an important criterion for analysing a problem using Spearman's correlation analysis.



(a) Spearman's correlation matrix-MCDO_ADF



(b) Spearman's correlation matrix-DER

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[2] Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep evidential regression. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[3] Fabio Arnez, Huáscar Espinoza, Ansgar Radermacher, and François Terrier. A comparison of uncertainty estimation approaches in deep learning components for autonomous vehicle applications. *arXiv preprint arXiv:2006.15172*, 2020.

[4] Jonathan T. Barron. A general and adaptive robust loss function. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4331–4339, 2019.

[5] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. 2006.

[6] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

[7] Yukun Ding, Jinglan Liu, Jinjun Xiong, and Yiyu Shi. Revisiting the evaluation of uncertainty estimation and its application to explore model complexity-uncertainty trade-off. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 22–31, 2020.

[8] Lorry Dysert. Is estimate accuracy an oxymoron. *Cost engineering*, 49(1):32–36, 2007.

[9] Di Feng, Lars Rosenbaum, and Klaus Dietmayer. Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3266–3273, 2018.

[10] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In *ICML'16 Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 1050–1059, 2016.

[11] Jochen Gast and Stefan Roth. Lightweight probabilistic deep networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3369–3378, 2018.

[12] Paul W. Goldberg, Christopher K. I. Williams, and Christopher M. Bishop. Regression with input-dependent noise: A gaussian process treatment. In *Advances in Neural Information Processing Systems 10*, volume 10, pages 493–499, 1997.

[13] Ziv Goldfeld and Yury Polyanskiy. The information bottleneck problem and its applications in machine learning. *IEEE Journal on Selected Areas in Information Theory*, 1(1):19–38, 2020.

[14] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR 2015 : International Conference on Learning Representations 2015*, 2015.

[15] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, pages 1321–1330, 2017.

[16] Michael I. Jordan. The conjugate prior for the normal distribution, 2010. URL `https://people.eecs.berkeley.edu/~jordan/courses/260-spring10/lectures/lecture5.pdf`. Accessed on: 2021-01-12. [Online].

[17] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision. In *NIPS'17 Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume 30, pages 5580–5590, 2017.

[18] Anoop Korattikara, Vivek Rathod, Kevin Murphy, and Max Welling. Bayesian dark knowledge. In *NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, volume 28, pages 3438–3446, 2015.

[19] Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International Conference on Machine Learning*, pages 2796–2804, 2018.

[20] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *ICLR (Poster)*, 2016.

[21] Alexey Kurakin, Ian J. Goodfellow, Samy Bengio, Yinpeng Dong, Fangzhou Liao, Ming Liang, Tianyu Pang, Jun Zhu, Xiaolin Hu, Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, Alan L. Yuille, Sangxia Huang, Yao Zhao, Yuzhe Zhao, Zhonglin Han, Junjiajia Long, Yerkebulan Berdibekov, Takuya Akiba, Seiya Tokui, and Motoki Abe. Adversarial attacks and defences competition. *arXiv preprint arXiv:1804.00097*, pages 195–231, 2018.

[22] Alex Labach, Hojjat Salehinejad, and Shahrokh Valaee. Survey of dropout methods for deep neural networks. *arXiv preprint arXiv:1904.13310*, 2019.

[23] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30, pages 6402–6413, 2017.

[24] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

[25] Antonio Loquercio, Ana I. Maqueda, Carlos R. del-Blanco, and Davide Scaramuzza. Dronet: Learning to fly by driving. *IEEE Robotics and Automation Letters*, 3(2):1088–1095, 2018.

[26] Antonio Loquercio, Mattia Segu, and Davide Scaramuzza. A general framework for uncertainty estimation in deep learning. *IEEE Robotics and Automation Letters*, 5(2):3153–3160, 2020.

[27] Andrey Malinin and Mark Gales. Predictive uncertainty estimation via prior networks. In *NIPS'18 Proceedings of the 32nd International Conference on Neural Information Processing Systems*, volume 31, pages 7047–7058, 2018.

[28] Andrey Malinin, Sergey Chervontsev, Ivan Provilkov, and Mark J. F. Gales. Regression prior networks. *arXiv preprint arXiv:2006.11590*, 2020.

[29] Senthil Mani, Anush Sankaran, Srikanth Tamilselvam, and Akshay Sethi. Coverage testing of deep learning models using dataset characterization. *arXiv preprint arXiv:1911.07309*, 2019.

[30] Simon J. D. Prince. *Computer Vision: Models, Learning, and Inference*. 2012.

[31] Qing Rao and Jelena Frtunikj. Deep learning for self-driving cars: chances and challenges. In *2018 IEEE/ACM 1st International Workshop on Software Engineering for AI in Autonomous Systems (SEFAIAS)*, pages 35–38, 2018.

[32] Andrew Michael Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan Daniel Tracey, and David Daniel Cox. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, 2019.

[33] Peter Schulam and Suchi Saria. Can you trust this prediction? auditing pointwise reliability after learning. In *22nd International Conference on Artificial Intelligence and Statistics, AISTATS 2019*, pages 1022–1031, 2019.

[34] Murat Sensoy, Lance M. Kaplan, and Melih Kandemir. Evidential deep learning to quantify classification uncertainty. In *Advances in Neural Information Processing Systems*, volume 31, pages 3179–3189, 2018.

[35] Burr Settles. *Active Learning*. 2012.

[36] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[37] Udacity. End-to-end-steering-angle-dataset, 2018. URL `https://github.com/udacity/self-driving-car/tree/5d438a227ba58cb8fb9facc36ce7de9a92ac1b63`. Accessed on: 2021-01-12. [Online].

[38] Aki Vehtari, Andrew Gelman, and Jonah Gabry. Practical bayesian model evaluation using leave-one-out cross-validation and waic. *Statistics and Computing*, 27(5):1413–1432, 2017.

[39] Yazhou Yang and Marco Loog. Active learning using uncertainty information. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pages 2646–2651, 2016.

[40] Jiayu Yao, Weiwei Pan, Soumya Ghosh, and Finale Doshi-Velez. Quality of uncertainty quantification for bayesian neural network inference. *arXiv preprint arXiv:1906.09686*, 2019.

[41] Lingxue Zhu and Nikolay Laptev. Deep and confident prediction for time series at uber. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 103–110, 2017.