A Literature Survey on Estimating Uncertainty in Deep Learning Models: Ensuring safety in Intelligent Systems

Soja Salim
SCT College of Engineering, APJ Abdul Kalam Technological University, Trivandrum, India, soja@sctce.ac.in

Dr.Jayasudha JS
Central University of Kerala, Kasargod
jayasudhajs@cukerala.ac.in

2023 2nd International Conference on Computational Systems and Communication (ICCSC) | 978-1-6654-9393-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICCSC56913.2023.10143025

***Abstract*— Popular Deep learning models suffer many drawbacks such as making wrong predictions with great confidence, lack of uncertainty estimation capability, and failure in real-time scenarios. The main reason for the uncertainty is due to the large gap between how neural networks are trained in practice and how they are evaluated in deployment. When it comes to safety-critical applications, it is very important to build confidence in the output that is obtained. A well-calibrated uncertainty quantification method can tell whether a model is confident in its predictions or not. This survey focuses on techniques used for uncertainty quantification in deep learning.**

Keywords— ***Uncertainty Quantification, Bayesian techniques, Non-Bayesian techniques, safety-critical applications, Calibration, Monte Carlo.***

1.Introduction

With the recent advances in AI, many crucial real-world applications rely on it. Deep learning models take many decisions in safety-critical applications such as autonomous driving or medical diagnosis. If any failures happen in such applications, then they will be damaging and effects the lives around them. So it is very important to know whether the decision is correct or not. Training a neural network is not only for making predictions but also to tell whether the model is confident in its output or not.

It is required to develop an approach for a neural network to crunch data, and output, not just a prediction but also the model's confidence level based on the quality of the available data. A high-performance model is not good enough, but it is also important to understand when the model cannot be trusted. If a model can detect when it is not confident in its predictions, it can sometimes save lives as well as money.

Using this neural network, predictions P(y|x) for specific benchmarks y given new x are possible. When it predicts wrong decisions, it may lead to the failure of the system. So, in order to avoid failures, it is necessary to find why, when, and, how a model becomes uncertain. The answer to "why" is that [1] the distribution of training datasets is different from that in real-world settings. The following are the reasons for the data set shift [2]

Simple covariate shift is when only the distributions of covariates x change and nothing else does.

Prior probability shift is when only the distribution over y changes and nothing else does.

Sample selection bias is when the distributions vary as a result of an unidentified sample rejection mechanism.

Imbalanced data is a planned dataset shift used for computational or modeling convenience.

Domain shift entails modifications to measurements.

Source component shift involves changes in the contribution components' strengths.

The answer to "when" is that in real-time applications, the model may behave indifferently from that of the training time. If such models are used in safety-critical applications it may lead to catastrophic effects. The answer to "how" is that uncertainty happens in two axes of neural networks [3] they are (i) Aleatoric uncertainty and (ii) epistemic uncertainty. The first one is the uncertainty in the data itself. This is also known as irreducible [4] uncertainty, since we collect a large amount of data then also we have underlying noise in the collection process that is inherent in the data itself. The only way to reduce aleatoric uncertainty is to change the sensor and get more accurate data. Epistemic uncertainty estimation is much more challenging as compared to that of aleatoric uncertainty and there are some emerging approaches to determine epistemic uncertainty.

If we collect a large amount of data, then also we have underlying noise in the collection process that is inherent in the data itself. The only way to reduce aleatoric uncertainty is to change the sensor and get more accurate data. Epistemic uncertainty is much more challenging to estimate than aleatoric uncertainty and there are some emerging approaches to determine epistemic uncertainty.

The most important reason for the failure of a system is the difference in real-time and training-time behaviors. This is due to the fact that the data available during real-time is different from that of the training time. Out-of-distribution data may lead to wrong predictions and it affects the entire safety of the system [5]. So most of the literature, researchers try to improve the model by applying different transitions to the data set like shifts in intensity, rotation and applying perturbations. The major gap identified in this topic is there are no standards or regulations to be followed in their evaluation and no discussed steps or measures to be followed in the real-time execution of the model.

This survey examines some of the recent approaches for estimating uncertainty, its effectiveness, and its

shortcomings. Also, try to explore the works that deal with the real-time evaluation of the model.

2 Uncertainty Estimation Using Bayesian techniques

There are many drawbacks to neural networks. Many hyperparameters require specific tuning and it takes large datasets or a huge amount of time. The Neural Networks trained with the best hyperparameters obtain point estimates of the weights in the network as in Fig1.a.

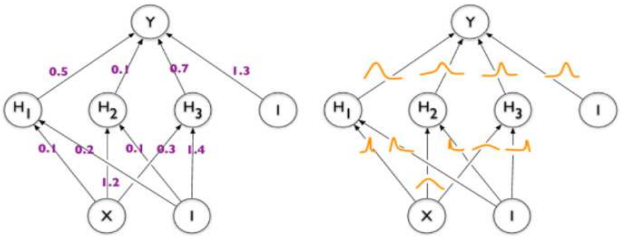

Fig. 1a. Deterministic NN

Fig. 1b. Bayesian NN

There is no uncertainty in this point estimates and this is very important when the outputs from the deep learning models do something pretty serious like medical diagnosis or they are operating self-driving cars.

2.1 Bayesian Learning

The Bayesian framework as in Fig. 1b, provides an effective mechanism to deal with uncertainty. Given a parameterized model f_θ, the data uncertainty in Bayesian modeling is formalized as a probability distribution over the model outputs y, and the model uncertainty is defined as a probability distribution over the model parameters θ. The prediction distribution over y∗, is described by Gal et. al. [6] and it is given in equation 1.

$$p(y^* \mid x^*, D) = \int \underbrace{p(y^* \mid x^*, \theta)}_{\text{Data}} \underbrace{p(\theta \mid D)}_{\text{Model}} d\theta \tag{1}$$

The uncertainty on the parameter estimation given a training sample set D is described by the term p(θ|D), which is also known as the posterior distribution of the model parameters. This is in general not tractable. Thinking of training neural networks as an inference problem, Bayes' theorem can be written as equation 2

$$p(\theta \mid D) = \frac{p(D \mid \theta)p(\theta)}{p(D)} \tag{2}$$

The prior distribution, p(θ) only gives information about θ. The likelihood estimation of data in D, p(D|θ) is the predicted distribution by the model parameter θ. For higher-dimensional data, the computation of the posterior probability is practically unrealizable. Many approaches like Monte Carlo dropout, Markov Chain Monte Carlo , and Variational Inference methods can be used to overcome the above-mentioned difficulty.

2.2 Monte Carlo (MC) dropout

This is a variation of the dropout layer [6] because of the Monte Carlo simulation this method has better results and accuracy. In this approach, a dropout layer is added to the model during each iteration based on the dropout ratio. Finally, an ensemble approach is used to approximate the mean prediction and it is given by equation 3.

$$y^* \approx \frac{1}{N}\sum_{i=1}^{N} y_i^* = \frac{1}{N}\sum_{i=1}^{N} f_{\theta_i}(x^*) \tag{3}$$

Where $f_{\theta 1}$,....., $f_{\theta N}$ are the expected value from N deterministic networks, parameterized by N samples, θ_1, θ_2, ..., θ_N.

The MC dropout method has been successfully applied in different applications. The main advantage of this method is that it can be directly applied to any existing architecture without any change.

2.3 Markov chain Monte Carlo (MCMC)

It is a class of algorithms for sampling from unknown posterior distributions. A Markov chain of samples' equilibrium distribution should be equal to the target distribution. Each Markov step is a sampling from the target distribution. If more steps are selected, that means more samples are chosen and there will be a more accurate assessment of the distribution. This approach relies on Markov screening which means the state of the sample t+1 only depends on the previous sample t. That is the conditional probability of a current sample which depends on the conditional probability of the previous sample and not on the history.

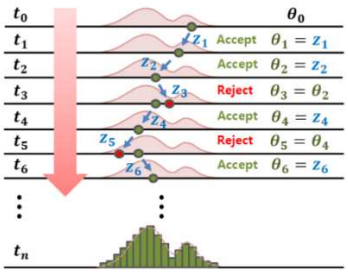

Fig. 2. Illustration of Markov Chain Monte Carlo Method

The MCMC algorithm generates the proposed sample ($\boldsymbol{z}$) from the current sample ($\boldsymbol{\theta i}$). The proposed sample [7] is either accepted ($\boldsymbol{\theta i} \rightarrow \boldsymbol{z}$) or rejected ($\boldsymbol{\theta i} \rightarrow \boldsymbol{\theta i}$) as shown in Fig 2.

This methodology can be applied when it is not possible to get to the distribution directly but it is required to construct a set of samples to search over the range of possible outcomes. The distributions in high dimensional spaces are very complicated and it is difficult to find the regions of high probability. This method is very useful in such cases. For sampling in higher dimensional space, this method

requires more samples and convergence takes more time.

2.4 Variational Inference

This method helps to approximate the posterior probability of unobserved variables [8]. Rather than finding the true posterior P(w|X,Y), this method approximates variational distribution q(w) parameterized by θ. Since it is derived from the observed data points X rather than being directly observed, the parameter is also known as a latent variable. Finding the combination of latent variables θ that can best represent the obtained result is the goal of the optimization process. The optimization objective is the Kullback-Leibler (KL) divergence, also known as relative entropy [9], which is used to assess how similar the two distributions are.

The KL divergence between the two distributions are given in equation 4

$$KL\left(q(\omega) \parallel (P(\omega \mid X,Y)\right) = \int q(\omega)\log\left(\frac{q(\omega)}{P(\omega \mid X,Y)}\right)d\omega$$
$$= \int q(\omega)\log(q(\omega))d\omega - \int q(\omega)\log(P(\omega \mid X,Y))d\omega$$
$$= \int q(\omega)\log(q(\omega))d\omega - \int q(\omega)\log(P(\omega,X,Y))d\omega + \log(P(X,Y)) \quad (4)$$

3 Uncertainty Estimation Using Non-Bayesian Techniques

The main idea behind this technique is to combine the different probability decisions or beliefs. There are different Non-Bayesian approaches are there. Some of the important techniques include Deep ensembles, Softmax calibration and Selective classification.

3.1 Deep ensembles

In traditional classification, we have many data points and each data point has features x, where x is some D dimensional data and y is the label. The classification technique uses the value of x to predict y. If a deep neural network is used for this task, it would simply characterize the function for the classification. In the Ensemble model, it will take a dataset and simply train multiple different deep neural networks and for the classification task, the new data point is given to all multiple models and finally aggregate the results from all models. Hence a better prediction is obtained than that of a single model. So this is called an ensemble model. If the ensemble members are deep neural networks then it is called a deep ensemble. By using the Ensemble model, a generalizable function can be learned [7] and is shown in Fig. 3

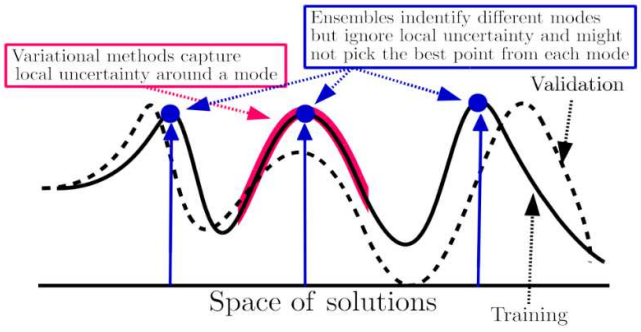

Fig. 3.Cartoon Illustration of the hypothesis deep ensembles explore diverse modes in function space

3.2 Softmax calibration

Calibration measures how well the predicted confidence aligns with the observed accuracy. The confidence is actually the probability estimation of the model by itself. The calibration can be realized using equation 5.

$$\mid P(\hat{Y} = y \mid \hat{p} = p) = p, \forall p \in [0,1] \quad (5)$$

The above equation is based on the classifier having data sets belonging to any of the classes, Y=1,2,…..,k and the classifier predicts $\hat{Y} = y$ for the given data x in X with a confidence $\hat{p}$ =p where p means the probaility of correctness.

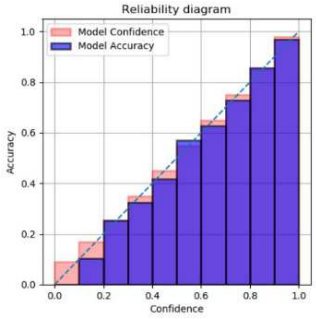

Fig. 4a. Calibrated Classifier

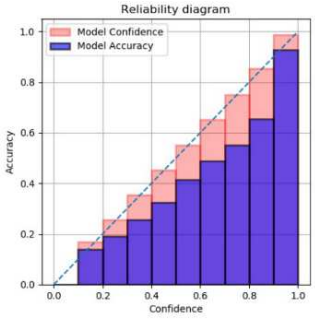

Fig. 4b. Uncalibrated Classifier

From Fig. 4a. and 4b, it is very clear how a well-calibrated classifier performs as compared to an uncalibrated classifier. Here the model confidence for each of the samples is put into multiple bins and this is plotted on the X-axis and the actual accuracy of the model is shown on the Y-axis. If it is perfectly calibrated then all the points should end up in the diagonal line. If the plot between confidence and accuracy is linear, then it will be an ideal calibration line. The calibration error is the degree to which the calibration curve for the classifier deviates from the ideal calibration line. The softmax function is used frequently to express per-class model prediction confidence since it transforms the logit values into a number between 0 and 1

3.3 Selective classification

This is an important tool in which the model can abstain from its prediction if it is not sure about the output. So, this is also known as the reject option [10] as it is very helpful in reducing the misclassification risk. This is very useful for critical applications when taking important decisions so humans can take control of it. A classifier f: X → Y [11], defined over the input X having m attributes and output Y having k class labels, the misclassification risk of a selective classifier is given in equation 6.

$$\hat{r}(f, g_\theta) = \frac{\frac{1}{m}\sum_{i=1}^{m} l(f(x_i), y_i) g_\theta(x_i)}{\frac{1}{m}\sum_{i=1}^{m} g_\theta(x_i)} \quad (6)$$

Here a selection function $g_\theta(x)$ is used , which is responsible for making the classifier to take decision on a particular data point.

$$g_\theta(x) = \begin{cases} 1 & \text{if } \kappa_f(x) \geq \theta \\ 0 & \text{otherwise.} \end{cases} \quad (7)$$

Table 1: Summary of different Uncertainty techniques and applications

Authors	Year	Architecture	Uncertainty techniques	Application
Catak et al.	2021	YoLo, SSD300 and SSD512	MC dropout	Object Recognition
Milanés et al.	2021	Shallow Convolutional Neural Network (SCNN-MCD) and an ensemble model (E-SCNN-MCD).	Ensemble	3D object detectors
Stoean et al.	2020	CNN-LSTM	MC dropout	Spinocerebellar ataxia type 2
Kendall et al.	2017	DenseNet architecture	MC dropout	Computer vision
Deepshikha et al.	2021	YOLOv5	MC DropBlock	Object Recognition
Lakshminarayanan et al.	2017	DNN	Deep Ensembles	Experiment on different datasets like MNIST, SVHN ,ImageNet etc.
Zhou et al.	2020	CNN	Markov Chain Monte Carlo	Contaminant Source Identification
Geifman et al.	2017	DNN	Selective classification	Experiment on different datasets like CIFAR-10, CIFAR-100 and ImageNet

In the equation 7 the function, $\kappa_f(x)$ gives the confidence score for the final prediction. This gives a way to use different confidence functions from MC dropout, ensemble, or any other methods discussed above. Here the data points whose confidence score is above the threshold θ are considered for classification.

4. Observations

The approaches discussed above provide different mathematical models for quantifying uncertainty in the predictions. Among the Bayesian techniques, the MC dropout approach provides better results in an efficient way. Also, the Variational Inference approach can overcome the difficulty of Bayesian learning in finding the posterior distribution. MCMC method is very useful for finding distributions in higher dimensional space.

In the case of Non-Bayesian Techniques, Deep ensemble approach increase the reliability of the predictions by combining the outputs from different neural networks. Predictions with high uncertainty (or low confidence) can be ignored in the presence of a correctly calibrated model to prevent unneeded model errors. Selective classification with the help of a selection function $g_\theta(x)$ helps to reduce the misclassification risk.

5. Conclusions

The different methods used to capture uncertainty in deep learning models are discussed. Neural networks do not have the inherent capability to perform uncertainty quantification. So it is very important for the developers to design models that not only give the prediction but also tell when it is not sure about the output which means uncertainty. Researchers use these methods as an offline procedure. Very few works explained the real-time or application phase scenario of these methods. When thinking about safety-critical applications, real-time uncertainty prediction is very important. Reliable and robust DL models can be built through proper uncertainty quantification only.

References

[1] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. Dillon, B. Lakshminarayanan, and J. Snoek, "Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift", Advances in Neural Information Processing Systems, 2019, pp. 13 991–14 002

[2] Quinonero-Candela J, Sugiyama M, Schwaighofer A, Lawrence ND, editors, "Dataset shift in machine learning",Mit Press; 2008 Dec 12.

[3] E. Hullermeier and W. Waegeman, "Aleatoric and epistemic ¨ uncertainty in machine learning: A tutorial introduction," arXiv preprint arXiv:1910.09457, 2019.

[4]Amini A, Schwarting W, Soleimany A, Rus D "Deep evidential regression", Advances in Neural Information Processing Systems. 2020;33:14927-37.

[5] Cai, Feiyang, and Xenofon Koutsoukos. "Real-time out-of-distribution detection in learning-enabled cyber-physical systems", *ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*,IEEE, 2020.

[6] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning",International conference on machine learning, 2016, pp. 1050–1059.

[7] Fort S, Hu H, Lakshminarayanan B," Deep ensembles: A loss landscape perspective", arXiv preprint arXiv:1912.02757. 2019 Dec 5.

[8] Iswariya Manivannan ,“A Comparative Study of Uncertainty Estimation Methods in Deep Learning-Based Classification Models”, Technical Report/Hochschule Bonn-Rhein-Sieg University of Applied Sciences, in 2020

[9] Ledermann S. Kullback S,”Information Theory and Statistics”,Population. 1962;17(2):377-8.

[10] Geifman Y, El-Yaniv R,”Selectivenet: A deep neural network with an integrated reject option”, In International conference on machine learning 2019 May 24 (pp. 2151-2159). PMLR.

[11] Yonatan Geifman and Ran El-Yani,” Selective classification for deep neural networks”, Advances in Neural Information Processing Systems, pages 4878–4887, 2017

[12] Catak FO, Yue T, Ali S. “Prediction surface uncertainty quantification in object detection models for autonomous driving”, IEEE International Conference on Artificial Intelligence Testing (AITest),2021 Aug 23 (pp. 93-100)

[13]Milanés-Hermosilla D, Trujillo Codorniú R, López-Baracaldo R, Sagaró-Zamora R, Delisle-Rodriguez D, Villarejo-Mayor JJ, Núñez-Álvarez JR,”Monte carlo dropout for uncertainty estimation and motor imagery classification”, Sensors. 2021 Oct 30;21(21):7241.

[14] Stoean C, Stoean R, Atencia M, Abdar M, Velázquez-Pérez L, Khosravi A, Nahavandi S, Acharya UR, Joya G. “Automated detection of presymptomatic conditions in Spinocerebellar Ataxia type 2 using Monte Carlo dropout and deep neural network techniques with electrooculogram signal”,Sensors. 2020 Jan;20(11):3032.

[15]Kendall A, Gal Y.,”What uncertainties do we need in bayesian deep learning for computer vision?”,Advances in neural information processing systems,2017;30.

[16]Deepshikha K, Yelleni SH, Srijith PK, Mohan CK.,”Monte carlo dropblock for modelling uncertainty in object detection”, arXiv preprint arXiv:2108.03614,2021 Aug 8.

[17] Lakshminarayanan, Balaji, Pritzel, Alexander, and Blundell, Charles,”Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles”, 2017. arXiv: 1612.01474 [stat.ML].

[18]Zhou Z, Tartakovsky DM,”Markov chain Monte Carlo with neural network surrogates: application to contaminant source identification”, Stochastic Environmental Research and Risk Assessment,2021 Mar;35(3):639-51.