Laplace Approximation Based Epistemic Uncertainty Estimation in 3D Object Detection

Anonymous Author(s)
Affiliation
Address
email

Abstract: Understanding the uncertainty of predictions is an admirable feature for perceptual modules in critical robotic applications. Laplace approximation provides a way to measure epistemic uncertainty. However, 3D object detectors have dense high-dimensional output space and indifferentiable operations, which makes it infeasible to directly apply existing Laplace-approximation-based approaches. In this work, we tailor Laplace approximation for 3D object detection, and propose solutions in Fisher approximation, Bayesian inference, and weight prior determination. The experiment results on the KITTI dataset empirically validate the effectiveness of our proposed methods, and demonstrate that Laplace approximation performs better uncertainty quality than Monte-Carlo Dropout and deterministic models.

Keywords: Laplace approximation, epistemic uncertainty, 3D object detection

1 Introduction

Detecting key objects in 3D space is an important module in robotic applications. It provides both semantic and spatial information for decision-making. Due to the success of deep learning, existing detectors have got great performance on benchmarks [1–3]. However, robots directly interact with the real world, the data distribution of which may drift over time and space. It is essential for a robotic perceptual module to know when it is uncertain, since the consequences of mistakes can be fatal in critical applications, like autonomous driving.

Laplace approximation [4] provides principled uncertainty estimation as other Bayesian approaches [5–10], and computes posterior weight distributions in terms of local curvatures. It can convert a deterministic model, trained under the maximum-a-posteriori framework, into a Bayesian model for uncertainty estimation, which is in a post-hoc way without changing the training procedure. Recent research reduces the computational costs on computing local curvatures with various approximations [11–15], and makes Laplace approximation applicable in modern neural networks.

However, existing methods focus on classification networks, which are commonly end-to-end and have a low-dimensional output space. Some of them are infeasible to use in detection cases. For instance, the Generalized-Gauss-Newton (GGN) based Fisher calculation [16] is impractical in detection cases since the output space of detectors is extremely high. Beside, some two-stage detectors rank and select proposals [17, 18], which are indifferentiable operations. It is not recommended to conduct Bayesian inference as the methods adopted for end-to-end networks.

In this paper, we tailor Laplace approximation methods to estimate epistemic uncertainty for 3D object detection under the context of autonomous driving. In specific, (1) we analyze the dense high-dimensional output space in 3D object detection, and suggest a Monte-Carlo approach to evaluate the Fisher information matrix. (2) To apply it in two-stage detectors, we propose an Uncertainty Separation and Aggregation method. It separates the uncertainty estimation in Region Proposal Network (RPN) and R-CNN stages and aggregates them in the final predictive distribution. (3) We also propose a heuristic method to determine weight distribution priors adaptively for each layer. The proposed Laplace-approximation method can easily convert an off-the-shelf 3D object detectors into a Bayesian neural network that is capable of estimating epistemic uncertainty. We evaluate various Laplace approximation implementations and compare their performance on the KITTI dataset. The experiment results show that the Laplace approximation performs consistently better uncertainty quality than deterministic models and Monte-Carlo Dropout baselines.

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

2 Related Works

2.1 Laplace approximation

The major challenge in applying Laplace approximation [4] to large-scale neural networks is its expensive computation on the Hessian matrix. Even though the Fisher information matrix can be its substitution, the exact Fisher requires storing and manipulating an $O(N^2)$ matrix, which is impractical to deploy in modern neural networks with the limited computational resources on mobile robots. Diagonal Fisher approximation is the lightest weight approach to approximate the Hessian. It ignores the correlation among model parameters, and largely reduces the Fisher matrix size into $O(N)$, which has been well applied to second-order optimization [19] as well as incremental learning [20]. To balance the curvature information and computational efficiency, some researchers approximate the exact Fisher matrix with Kronecker Factorized Fisher [15] and low-rank approximations [14]. Some work applied Laplace approximation to a subset of model parameters [11–13]. All of these works focus on end-to-end classification networks. In contrast, we consider the 3D object detection task. The network architecture of a detector is more complicated than common image classifiers. It may have a high-dimensional dense output space and contain indifferentiable operations, like ranking and proposal selection.

2.2 LiDAR-based 3D object detection

The geometrical information in LiDAR scans can be used to classify and locate key objects in 3D space. Compared to 2D boxes, 3D boxes are more adorable in path planning for mobile robots. Most work can be categorized into Region-Proposal-Network-based (RPN-based) and R-CNN-based methods. RPNs are one-stage networks which consume LiDAR scans and estimate 2D maps or 3D volumes for classification and regression [21–23]. To improve computational efficiency, Yan et al. adopted sparse convolution operations to replace the conventional dense convolution layers [22]. R-CNN-based methods are two-stage networks[1] [17, 18, 24], the first stage of which extracts features and generates 3D proposals. All the generated proposals are ranked in the second stage, and the top proposals are selected for further refinement. The ranking and selection operations are indifferentiable, raising challenges for the Monte-Carlo-based Bayesian inference, since the order of proposals may change if the model parameters changes, as illustrated in Figure 2(top). We propose an uncertainty-separation-and-aggregation method to enable the use of Laplace approximation in RCNN-based 3D object detection networks, and adopt PointRCNN [17] as an instance to demonstrate its usefulness.

2.3 Uncertainty estimation in object detection

Some recent works estimate uncertainties in robotic perceptual modules. Hall et al. extended object detection to probabilistic object detection and quantified spatial and semantic uncertainties of detections with Monte-Carlo Dropout [25]. They proposed the Probabilistic-based Detection-quality (PDQ) to assess the estimated predictive distribution with deterministic detection annotations. The idea of probabilistic object detection was further extended to 3D object detection [26–28]. Feng et al. [26, 27] made an RPN-based detector PIXOR [21] to estimate aleatoric uncertainties by adding an additional regression layer, as [29]. The estimated uncertainties were evaluated independently for each output dimension with expected calibration errors [30, 31]. To evaluate the general predictive distribution for LiDAR-based 3D object detection, Wang et al. [28] extended the idea of PDQ and proposed mAP(JIoU) which evaluates the differences between the estimated predictive distribution and a ground-truth predictive distribution. They proposed a generative model to obtain the ground-truth predictive distribution. These works provide feasibility to evaluate predictive distribution in 3D object detection. In this work, we evaluate predictive distributions with mAP(IoU) [1], mAP(JIoU) [28] and expected calibration errors [26].

[1] In this paper, "one-stage detectors/networks" refers to the RPN-based 3D object detectors, and "two-stage" cases refer to the RCNN-based counterparts.

3 Methodology

We first provide preliminaries of epistemic uncertainty estimation. Given a dataset $\mathcal{D} = \{\mathbf{X}, \mathbf{Y}\} = \{(\mathbf{x}_i, \mathbf{y}_i)|i = 1, ..., N\}$, we have a neural network $f_\theta(\mathbf{x})$ with its prior weight distribution $p(\theta)$. We want to derive the posterior weight distribution $p(\theta|\mathbf{X}, \mathbf{Y})$, so that we can conduct Bayesian inference and get the predictive distribution with

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{y}^*|\mathbf{x}^*, \theta)p(\theta|\mathbf{X}, \mathbf{Y})d\theta, \tag{1}$$

where $p(\mathbf{y}^*|\mathbf{x}^*, \theta)$ depicts the predictive model. In classification tasks, it is commonly defined as a categorical distribution with $p(\mathbf{y}^c = 1|\mathbf{x}, \theta) = \text{softmax}^c(f_\theta(\mathbf{x}))$, where the superscript c denotes the $c-$th element of a vector. In regression tasks, it can be defined as $p(\mathbf{y}|\mathbf{x}, \theta) = \mathcal{N}(\mathbf{y}; f_\theta(\mathbf{x}), \tau^{-1}\mathbf{I})$. The τ can be a hyper-parameter or an estimation from data. This Bayesian inference equation catches the **epistemic uncertainty** caused by the weight distribution $p(\theta|\mathbf{X}, \mathbf{Y})$.

We can estimate $p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y})$ with the Monte-Carlo estimator by calculating the population moments as [32]. In the following, we denote this step as moment estimation. In classification,

$$p(\mathbf{y}^{*,c} = 1|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) \approx \frac{1}{T} \sum_{\theta \in P(\theta|\mathbf{X},\mathbf{Y})} \text{softmax}^c(f_\theta(\mathbf{x}^*)). \tag{2}$$

The mode can be estimated by $c^* = \arg\max_c p(\mathbf{y}^{*,c} = 1|\mathbf{x}^*, \mathbf{X}, \mathbf{Y})$, and the uncertainty can thus be quantified in terms of the $p(\mathbf{y}^{*,c^*} = 1|\mathbf{x}^*, \mathbf{X}, \mathbf{Y})$. In regression, the mean and covariance matrix of the predictive distribution can be estimated with:

$$\mathbb{E}_{y^*}[\mathbf{y}^*] = \frac{1}{T} \sum_{\theta \in P(\theta|\mathbf{X},\mathbf{Y})} f_\theta(\mathbf{x}^*) \tag{3}$$

$$\text{Cov}[\mathbf{y}^*] = \frac{1}{T} \sum_{\theta \in p(\theta|\mathbf{X},\mathbf{Y})} f_\theta^T(\mathbf{x}^*) f_\theta(\mathbf{x}) + \tau^{-1}\mathbf{I} - \mathbb{E}_{y^*}[\mathbf{y}^*]^T \mathbb{E}_{y^*}[\mathbf{y}^*] \tag{4}$$

Therefore, estimating the epistemic uncertainties in the predictive distribution $p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y})$ is converted into the problem of evaluating the posterior weight distribution $p(\theta|\mathbf{X}, \mathbf{Y})$.

3.1 Laplace approximation based epistemic uncertainty estimation

The logarithm posterior weight distribution $\log p(\theta|\mathbf{X}, \mathbf{Y})$ can be unfolded with the Bayes rule:

$$\log p(\theta|\mathbf{X}, \mathbf{Y}) = \log p(\mathbf{Y}|\mathbf{X}, \theta) + \log p(\theta) - \log p(\mathbf{Y}|\mathbf{X}), \tag{5}$$

where $\log p(\mathbf{Y}|\mathbf{X})$ is independent to θ. We assume θ is trained with the loss given by the negative log-probability and a regularizer, i.e.

$$L(\theta, \mathbf{X}, \mathbf{Y}) = -\log p(\mathbf{Y}|f_\theta(\mathbf{X})) + \lambda||\theta||, \tag{6}$$

where $|| \cdot ||$ can be a L1 or L2 norm[2]. This is the case for standard least-squared and cross-entropy objective functions, where $p(\mathbf{Y}|f_\theta(\mathbf{X}))$ is multivariate normal and multinomial distributions, respectively. Minimizing $L(\theta, \mathbf{X}, \mathbf{Y})$ is equivalent to maximizing the log-posterior $\log p(\theta|\mathbf{X}, \mathbf{Y})$[11].

Laplace approximation adopts a second-order expansion of $L(\theta, \mathbf{X}, \mathbf{Y})$ at the optimal point θ^* to construct a Normal distribution approximating $p(\theta|\mathbf{X}, \mathbf{Y})$, i.e.

$$\log p(\theta|\mathbf{X}, \mathbf{Y}) \approx \log p(\theta^*|\mathbf{X}, \mathbf{Y}) + \frac{1}{2}(\theta - \theta^*)^T \mathbf{H}_{\log p(\theta^*|\mathbf{X},\mathbf{Y})}(\theta - \theta^*), \tag{7}$$

where $\mathbf{H}_{\log p(\theta^*|\mathbf{X},\mathbf{Y})} = \frac{\partial^2}{\partial^2\theta} \log p(\theta^*|\mathbf{X}, \mathbf{Y})$ is the Hessian matrix of $\log p(\theta^*|\mathbf{X}, \mathbf{Y})$. By substituting (5) into this definition, we have $\mathbf{H}_{\log p(\theta^*|\mathbf{X},\mathbf{Y})} = \mathbf{H}_{\log p(\mathbf{Y}|\mathbf{X},\theta)} + \frac{\partial^2}{\partial^2\theta}[\log p(\theta)]$. The second term can be evaluated easily if $p(\theta)$ is simple or in a closed form. In the first term, the Fisher[3] $\mathbf{F}$ can be a good substitute for the Hessian matrix [15, 20, 33], which is defined as

$$\mathbf{F} \doteq \mathbb{E}_{P_\mathbf{X}}[\mathbb{E}_{P_{\mathbf{Y}|\mathbf{X},\theta}}[\nabla \log p(\mathbf{Y}|\mathbf{X}, \theta)^T \nabla \log p(\mathbf{Y}|\mathbf{X}, \theta)]] \tag{8}$$

$$= -\mathbb{E}_{P_\mathbf{X}}[\mathbb{E}_{P_{\mathbf{Y}|\mathbf{X},\theta}}[\mathbf{H}_{\log p(\mathbf{Y}|\mathbf{X},\theta)}]]. \tag{9}$$

[2] If $|| \cdot ||$ is a L2-norm, the corresponding $p(\theta)$ is a Normal distribution $\mathcal{N}(\theta; 0, \lambda^{-1}I)$. It will be a Laplace distribution, if $|| \cdot ||$ is a L1-norm.

[3] "The Fisher" is the shorthand of "the Fisher information matrix" in this paper.

Algorithm 1: Monte-Carlo-based approach to calculating the Fisher

Input : $\mathbf{X}$: inputs; f_θ: deterministic neural network;
M: number of Monte-Carlo samples;
Σ_0^{reg}: prior covariance matrix

Output : $\mathbf{F}$: the Fisher

initialize $\mathbf{F}$; $N \leftarrow len(\mathbf{X})$;
foreach $\boldsymbol{x} \in \boldsymbol{X}$ **do**
  $\mathbf{y}_0 \leftarrow f_\theta(\mathbf{x})$;
  foreach $m < M$ **do**
    $\tilde{\mathbf{y}}^{cls} \leftarrow$ sample from categorical$(\mathbf{y}_0^{cls})$;
    $\tilde{\mathbf{y}}^{reg} \leftarrow$ sample from $\mathcal{N}(\mathbf{y}_0^{reg}, \Sigma_0^{reg})$;
    $\mathbf{g} \leftarrow \nabla_\theta \log p(\tilde{\mathbf{y}}^{cls}, \tilde{\mathbf{y}}^{reg} | f_\theta(\mathbf{x}))$;
    $\mathbf{F} += \frac{1}{NM}\mathbf{g}^T\mathbf{g}$;
  end
end
return $\mathbf{F}$;

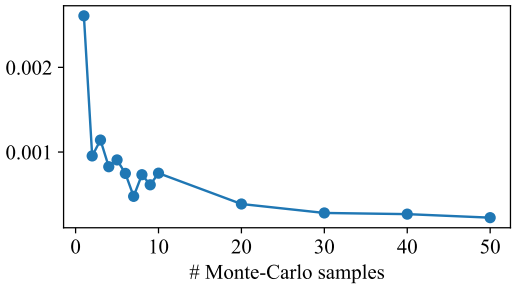


Figure 1: Mean absolute difference between the Fishers, computed with GGN and Monte-Carlo approaches (Algorithm 1). It is based on a Linear model on the MNIST dataset.

The approximate posterior weight distribution is $p(\theta|\mathbf{X}, \mathbf{Y}) \approx \mathcal{N}(\theta; \theta^*, \Sigma)$, where $\Sigma = [\mathbf{F} + \Sigma_0^{-1}]^{-1}$, if we consider a Gaussian prior. Applying Laplace approximation in the object detection is challenging: (1) due to the high-dimensional dense representations of its output space, GGN-based approaches are impractical to calculate the Fisher in the object detection; (2) due to indifferentiable operations, a two-stage object detector may overestimate uncertainty and wrongly predict the mode. We introduce our solutions for 3D object detection in Section 3.1.1 and 3.1.2, and also propose a heuristic approach to determine the priors of weight distribution in Section 3.1.3.

3.1.1 Evaluating the Fisher

To evaluate the Fisher[4], [16] suggests a Generalized Gauss-Newton method which adopts the split $\nabla_\theta \log p(\mathbf{y}|\mathbf{x}, \theta) = J_{f_\theta(\mathbf{x})}^T \nabla_{f_\theta(\mathbf{x})} \log p(\mathbf{y}|f_\theta(\mathbf{x}))$ and calculates the Fisher with

$$\mathbf{F} = \mathbb{E}_{P_\mathbf{X}}[J_{f_\theta(\mathbf{x})}^T \mathbb{E}_{P_{\mathbf{Y}|\mathbf{X},\theta}}[\nabla_{f_\theta(\mathbf{x})} \log p(\mathbf{y}|f_\theta(\mathbf{x}))^T \nabla_{f_\theta(\mathbf{x})} \log p(\mathbf{y}|f_\theta(\mathbf{x}))] J_{f_\theta(\mathbf{x})}], \tag{10}$$

where $J_{f_\theta(\mathbf{x})} \in \mathbb{R}^{O \times P}$ is the Jacobian of $f_\theta(\mathbf{x})$ w.r.t. θ, P is the parameter dimension, and O is the output space dimension. It is easy to analytically evaluate the inner expectation over $P_{\mathbf{Y}|\mathbf{X},\theta}$, if the output space dimension is not high and the loss functions are standard cross-entropy or least-square functions. However, the output space of 3D object detection is commonly in high-dimension dense representations, like 2D/3D maps [21–23], and point lists [17, 18]. It is computationally intractable to compute $J_{f_\theta(\mathbf{x})}$ in 3D object detection for evaluating the Fisher with (10).

To solve it, we adopt the Monte-Carlo method used in [33] to evaluate the Fisher in the 3D object detection. It samples $\mathbf{y}$ from the predictive distribution and then runs the back-propagation using the samples as targets. For the post-hoc Laplace approximation method in this paper, the predictive distribution here might be ill-defined. It is because we only have a deterministic model before calculating the Fisher. Thus, a prior predictive distribution must be introduced so that we can sample from it and compute the Fisher. Algorithm 1 illustrates this method of Fisher calculation. The result of Monte-Carlo-based approach converges to those of the GGN method when the number of samples increases, as shown in Figure 1.

3.1.2 Uncertainty Separation and Aggregation

To calculate the predictive distributions, we estimate moments of the predictive distribution with Monte-Carlo integration (2), (3), (4). It works well in one-stage detectors. However, a direct extension to two-stage detectors may fail as shown in Figure 2 (top). In two-stage detectors, RPN generates proposals, and RCNN ranks them as well as refines the selected proposals. Both ranking and selection are indifferentiable operations. They impede the gradient back-propagation in evaluating the Fisher. In addition, different weight samples drawn in Monte-Carlo sampling may generate different proposals. These proposals across different weight samples are probably not matched in numbers or orders. This mismatching finally causes wrong estimation of mode and uncertainties.

[4]We are discussing the standard Fisher (8), which is calculated with the expectation over the predictive distribution. It is distinguished from the "empirical Fisher", which is computed over the training distribution. The differences have been discussed in [16, 34].

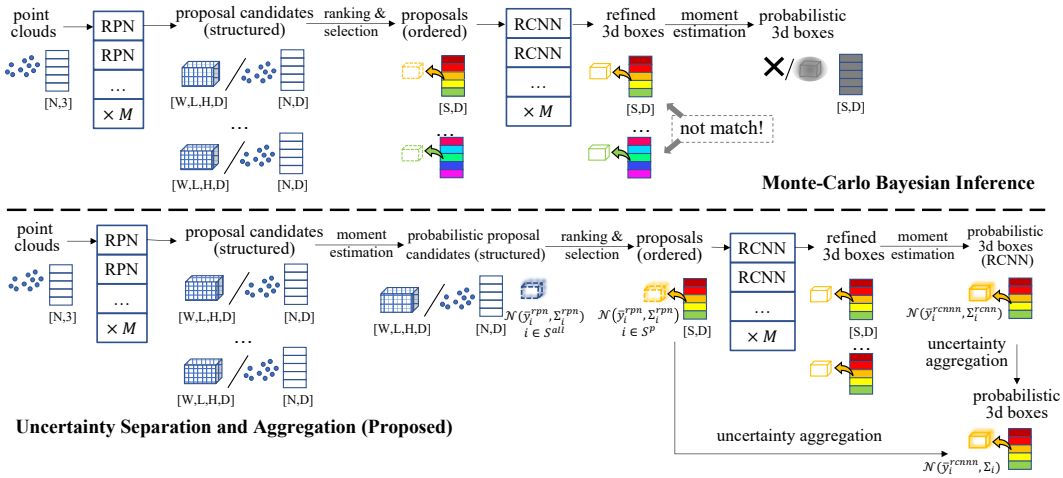


Figure 2: Top: Directly adopting the Monte-Carlo Bayesian Inference method in a two-stage detector. Bottom: Uncertainty separation and aggregation.

To solve this problem in two-stage detectors, we separate the uncertainty estimation for RPN and R-CNN, and aggregate them in the final predictive distribution as shown in Figure 2 (bottom). In estimating the uncertainties in RPN, we conduct moment estimation on proposal candidates, instead of the ordered proposals which have been ranked by classification probabilities. It is because the proposal candidates are in structured representations, like 2D/3D volumes [21–23], and point lists[17, 18, 24]. These structured representations are naturally matched among different Monte-Carlo samples. By moment estimation, we can get both the mode and uncertainty for each proposal candidate, and denote one as $\mathcal{N}(\mathbf{y}_i^{rpn}; \bar{\mathbf{y}}_i^{rpn}, \Sigma_i^{rpn}), i \in \mathcal{S}^{all}$, where $\mathcal{S}^{all}$ is the set containing all proposal candidates. Then we rank them in terms of their modes as conventional two stage detectors [17, 18], and select a subset $\mathcal{S}^p \subset \mathcal{S}^{all}$.

In estimating the uncertainties in R-CNN, we input the modes of probabilistic proposals $\{\bar{\mathbf{y}}_i^{rpn} | i \in \mathcal{S}^p\}$ to each Monte-Carlo samples. With the same input, the outputs of Monte-Carlo samples will be matched in the same order. By moment estimation, we get both modes and uncertainties for each refinement results, and denote one as $\mathcal{N}(\mathbf{y}_i^{rcnn}; \bar{\mathbf{y}}_i^{rcnn}, \Sigma_i^{rcnn}), i \in \mathcal{S}^p$. In aggregating the uncertainties of a proposal box from RPN and R-CNN, we keep the mode of $\mathbf{y}_i^{rcnn}$ since it is the refinement results of proposal $\mathbf{y}_i^{rpn}$. Besides, we accumulate the uncertainties of $\mathbf{y}_i^{rpn}$ and $\mathbf{y}_i^{rcnn}$, which can be considered as adding up uncertainties from two independent sources. The final predictive distribution is thus $\{\mathcal{N}(\mathbf{y}_i; \bar{\mathbf{y}}_i^{rcnn}, \Sigma_i^p + \Sigma_i^{rcnn}) | i \in \mathcal{S}^p\}$.

3.1.3 A heuristic approach to sampling from weight distributions

An important step in moment estimation with (2), (3), (4) is sample from the approximate weight distribution $p(\theta|\mathbf{X}, \mathbf{Y}) = \mathcal{N}(\theta; \theta^*, \Sigma)$, where $\Sigma = [\mathbf{F} + \Sigma_0^{-1}]^{-1}$. In practice, the corresponding elements of some parameters in the prior precision matrix Σ_0^{-1} must remain large to compensate for the intrinsic inaccuracy of Fisher approximation, but it has the side effect of washing out some small but meaningful values [33].

Through trial and error, we find a relatively simple and effective way to determine the elements in Σ_0 for each parameter in diagonal Fisher approximation. As illustrated in Algorithm 2, we compute a threshold t for each layer. All the parameters in each layer, whose Fisher values are smaller than t will be given an infinitely large value in Σ_0^{-1}. For the other parameters, their Σ_0^{-1} element is set as t. In our experiment, we empirically set t as the 30-quantile positive element in each layer. A parameter α is introduced to scaling the covariance matrix, and we found setting α proportional to t works well in practice. We compare the uncertainty estimation quality of our proposed heuristic approach to the naive method with a homogenous Σ_0 for all parameters in our experiments. Superior performance of the heuristic method can be seen by comparing the last two plots in Figure 3.

methods	PP				SC				PR			
	classification		regression		classification		regression		classification		regression	
	mean	std.	mean	std.	mean	std.	mean	std.	mean	std.	mean	std.
det.	0.115	-	-	-	0.100	-	-	-	0.113	-	-	-
mc-dropout	0.112	0.000	0.763	0.001	0.126	0.000	0.594	0.001	0.104	0.001	0.745	0.002
LL (emp.)	0.115	0.000	0.527	0.004	0.100	0.000	0.511	0.003	0.111	0.002	0.354	0.006
LM (emp.)	-	-	-	-	-	-	-	-	0.063	0.001	**0.313**	0.005
full (emp.)	**0.096**	0.000	0.424	0.003	**0.088**	0.000	0.392	0.001	**0.062**	0.001	0.566	0.020
LL (std.)	0.115	0.000	**0.310**	0.008	0.100	0.000	0.308	0.025	0.111	0.002	0.519	0.001
LM (std.)	-	-	-	-	-	-	-	-	0.099	0.002	0.479	0.002
full (std.)	0.114	0.000	0.327	0.008	0.095	0.001	**0.299**	0.023	0.074	0.001	0.436	0.041

Table 1: Expected calibration error (↓). The "det", "LL", "LM", and "full" are short for "deterministic", "last-layer", "last-module", and "full-net". The results are computed from 5 random seeds.

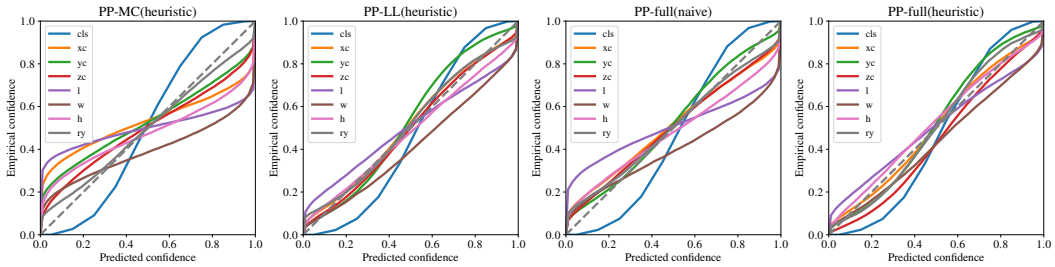


Figure 3: Calibration plots of PointPillar. The curves closer to the gray line (upper bound) represent better results. More plots are included in the Appendix.

4 Experiments

In this section, we conduct experiments on the KITTI dataset [1] to explore the following questions: (1) How is the quality of the estimated epistemic uncertainty? (2) Does the accuracy of mode estimation get affected after applying Laplace approximation? (3) How is the performance of the predictive distribution?

We adopt well-trained 3D object detectors as deterministic models and convert them with the Laplace-approximation-based approaches in Section 3. The adopted 3D detectors contain one-stage detectors PointPillar[23] ("PP") and Second[21] ("SC"), as well as a two-stage detector PointRCNN[17] ("PR"). In particular, we compute the Fisher on the training set and inference with it on the validation set. Ten Monte-carlo samples are adopted to approximate the integration in (1). Hyperparameters are tuned on a hand-hold set. We set $\alpha = 0.003$ for the standard Fisher, and $\alpha = 0.01$ for the empirical Fisher. For more implementation details, please refer to the Appendix.

Algorithm 2: Heuristic method to calculate the covariance of posterior weight distribution

Input : $\bar{\mathbf{F}}^l, l = 1, ..., M$: diagonal vector of the Fisher for the l-th layer; α: scaler of covariance matrices.

Output : $C^l, l = 1, ..., M$: diagonal vector of the covariance matrix of the parameters.

foreach $\bar{F}^l$ **do**

  initialize C^l with all zeros ;

  $t \leftarrow$ the 30-quantile positive element in $\bar{\mathbf{F}}^l$;

  $idx \leftarrow$ indices of $\bar{\mathbf{F}}^l > t$;

  /* $\mathbf{I}$ is an identity matrix */ ;

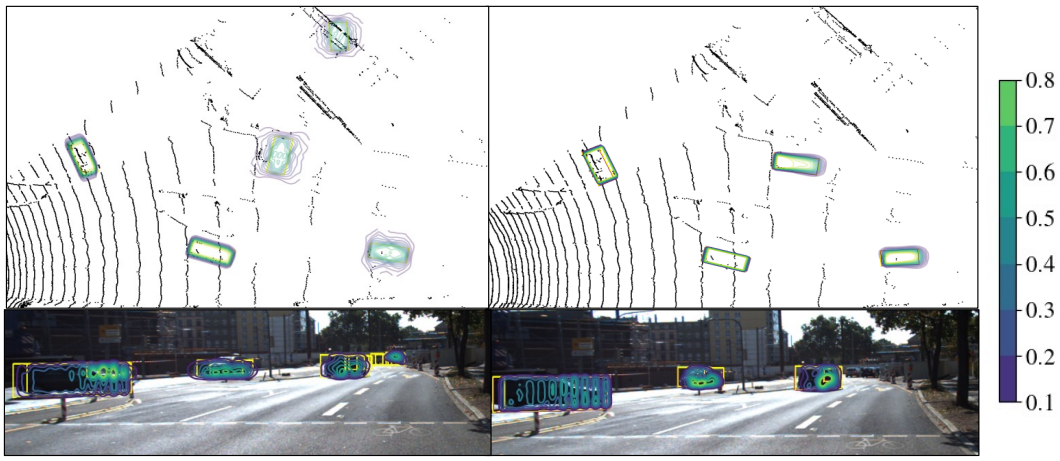
  $C^l[idx] \leftarrow \alpha t(\bar{\mathbf{F}}^l + t \times \mathbf{I})^{-1}$;

end

return $C^l, l = 1, ..., M$;

4.1 Quality of uncertainty estimation

A good uncertainty estimation should reflect the real empirical confidence in the data. For instance, all the 0.8-probability estimations of a classification model should have precision 80% on the given dataset. Then we say the estimated uncertainty is of good quality. Expected Calibration Error (ECE) [30, 31] is a good metric to evaluate the uncertainty quality, and a smaller value depicts better uncertainty quality. We treat each dimension independently, following [26], and compare the ECEs of classification and regression for the following baselines and various Laplace-approximation implementations:



(a) epistemic uncertainty (b) spatial uncertainty

Figure 4: Visualization of estimated epistemic uncertainty (a), and the ground-truth spatial uncertainty [28] used in evaluating mAP(JIoU). The contours reflect the probability of a point subject to an object (objectiveness probability).

- deterministic model: we use the softmax outputs as classification confidences.
- mc-dropout [32]: we add a dropout layer before the last convolution/linear layer to generate the Monte-Carlo-based predictive distribution. To make a fair comparison, we use the same model parameters as the others without retraining;
- full-net (empirical/standard): we implement Laplace approximation on all the layers with the diagonal approximation of the empirical/standard Fisher.
- last-layer (empirical/standard) [11]: Laplace approximation is applied on the last-layer only.
- last-module: Laplace approximation is applied on the R-CNN network for the two-stage detector.

Table 1 reports the ECE results, from which we can find that (1) The estimated quality of Monte-Carlo-based uncertainty estimation methods, including both Monte-Carlo Dropout and Laplace Approximation, are better calibrated than the deterministic model outputs. (2) Laplace approximation performs consistently better than Monte-Carlo Dropout in both classification and regression. (3) In most cases, the uncertainty quality of full-net Laplace approximation is better than sub-net Laplace approximation, except in the case of the empirical Fisher in PointRCNN. It is opposite to the finding in [11] which claims the last-layer empirical Fisher owns the best performance and full-net Fisher suffers from an under-fitting problem. We attribute our result to the fact that we implement the Laplace approximation with a diagonal approximation which contains much fewer parameters than the either full or K-FAC Fisher matrix used in [11].

We also report the calibration plots in Figure 3. It demonstrates the uncertainty quality separately for each dimension. It also shows that our proposed heuristic approach performs better uncertainty quality than the naive method if we compare the last two plots in Figure 3.

4.2 Accuracy of estimated mode

The covariance matrix of approximate posterior weight distribution is scaled by α in Algorithm 2. To explore its effect on the accuracy of the estimated mode, we plot the relationship between α and mAP(IoU). For better visualization, we compute the average mAP(IoU) over classes and different levels, denoted as $\mathbb{E}_{\text{class,level}}[\text{mAP(IoU)}]$ in Figure 5. It is found that as α increases, $\mathbb{E}_{class,level}[\text{mAP(IoU)}]$ decreases. It achieves its maximum at $\alpha = 0$, in which case the model is deterministic and cannot estimate uncertainty. The mAP(IoU) degrades sightly when the α is small, while an extremely large value of α will greatly deteriorate the mAP(IoU). It suggests adopting a relatively small value of α to obtain the uncertainty and preserve the mAP(IoU) performance.

4.3 Quality of predictive distributions

In this part, we evaluate the quality of predictive distributions with the metric mAP(JIoU)[28]. It is a metric similar to mAP(IoU) [1], but it calculates Jaccard IoU (JIoU) between two predictive distri-

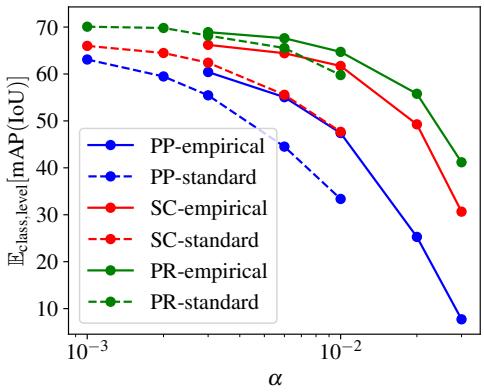


Figure 5: Relationship between α and $\mathbb{E}_{\text{class,level}}[\text{mAP(IoU)}]$ (↑) (full-net). The stars depict the results of the deterministic models.

		bev.			3d		
method	class	easy	mod.	hard	easy	mod.	hard
PP	car	82.1	79.8	76.9	45.9	29.6	25.1
	ped.	19.2	21.1	21.0	28.1	28.3	26.6
	cyc.	47.9	41.9	39.6	45.3	32.4	30.8
SC	car	86.4	84.3	80.9	57.1	36.9	30.7
	ped.	23.1	26.1	25.9	30.9	31.0	28.7
	cyc.	53.2	49.7	47.9	44.9	34.3	32.3
PR	car	82.9	80.8	77.0	42.7	31.1	27.1
	ped.	26.9	27.2	24.0	40.6	36.9	32.2
	cyc.	63.4	51.7	49.7	62.3	43.3	41.2

Table 2: mAP(JIoU) (↑)(standard Fisher, full-net) results on the KITTI val set .

butions. We generate the ground-truth predictive distribution from the generative model proposed in [28] with the 3D bounding box annotations and LiDAR observations. This generative model characterizes the generative process of key-object points within 3D bounding boxes. As shown in Figure 4 (b), we can observe lower objectiveness probability in the car-tail regions where the observed points are sparse. Figure 4 (a) shows the estimated epistemic uncertainty, which is small in the near regions and higher in far regions and for those false positive estimations. We also report the mAP(JIoU) results in Table 2. It shows that the mAP(JIoU) performance is relatively good in big objects, like cars and cyclists, and performs poorly in small objects, like pedestrians.

5 Limitations

The proposed method computes the Bayesian inference formula based on Monte-Carlo, It requires $M\times$ computational costs than a single forward computation. The real-time performance can be improved if it parallels the M Monte-Carlo computations with batched inputs [32]. For the robotic application with strong constraints on computational resources, the subnet implementation that only applies Laplace approximation on the last layer also provides good-quality uncertainty estimations, as shown in Table 1. This paper assumes that both weight distributions and predictive distributions conform to the Normal distribution. It is limited in characterizing the multi-modality. It can be partially alleviated by fitting Laplace approximation to each modality [35].

The metrics mAP(IoU) and mAP(JIoU) can evaluate the mode accuracy and general performance of predictive distribution so that we can tune hyperparameters and analyze their performance in different classes. However, they are not good metrics for comparison crossing different types of detectors, because a fair comparison requires aligning the weight perturbation at the same level. It is impractical to implement if their network structures are different. One possible solution is to compare the performance of downstream tasks with the estimated epistemic uncertainties as inputs, like active learning.

6 Conclusion

In this paper, we tailor Laplace approximation for 3D object detectors. An uncertainty-separation-and-aggregation method and a heuristic approach to weight prior determination are proposed to make it work in 3D object detection. The experiment results on the KITTI dataset show that the estimated epistemic uncertainty is of better quality than deterministic models and Monte-Carlo Dropout baselines. Full-net diagonal Laplace approximation, which provides good uncertainty quality, is recommended in off-line applications, like model diagnosis, active learning, and life-long learning. If the robotic applications have strong computational resource constraints, subnet Laplace approximation is a good balance between real-time performance and uncertainty quality.

References

[1] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.

[2] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[3] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov. Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

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

[5] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight uncertainty in neural network. In *International conference on machine learning*, pages 1613–1622. PMLR, 2015.

[6] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[7] D. P. Kingma, T. Salimans, and M. Welling. Variational dropout and the local reparameterization trick. *Advances in neural information processing systems*, 28, 2015.

[8] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J. M. Hernández-Lobato, and A. L. Gaunt. Deterministic variational inference for robust bayesian neural networks. In *International Conference on Learning Representations*, 2018.

[9] A. D. Cobb and B. Jalaian. Scaling hamiltonian monte carlo inference for bayesian neural networks with symmetric splitting. In *Uncertainty in Artificial Intelligence*, pages 675–685. PMLR, 2021.

[10] M. Welling and Y. W. Teh. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pages 681–688. Citeseer, 2011.

[11] E. Daxberger, A. Kristiadi, A. Immer, R. Eschenhagen, M. Bauer, and P. Hennig. Laplace redux-effortless bayesian deep learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[12] A. Kristiadi, M. Hein, and P. Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. In *International conference on machine learning*, pages 5436–5446. PMLR, 2020.

[13] E. Daxberger, E. Nalisnick, J. U. Allingham, J. Antorán, and J. M. Hernández-Lobato. Bayesian deep learning via subnetwork inference. In *International Conference on Machine Learning*, pages 2510–2521. PMLR, 2021.

[14] J. Lee, M. Humt, J. Feng, and R. Triebel. Estimating model uncertainty of neural networks in sparse information form. In *International Conference on Machine Learning*, pages 5702–5713. PMLR, 2020.

[15] H. Ritter, A. Botev, and D. Barber. A scalable laplace approximation for neural networks. In *6th International Conference on Learning Representations, ICLR 2018-Conference Track Proceedings*, volume 6. International Conference on Representation Learning, 2018.

[16] J. Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21:1–76, 2020.

[17] S. Shi, X. Wang, and H. Li. Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 770–779, 2019.

[18] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li. Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538, 2020.

[19] M. Khan, D. Nielsen, V. Tangkaratt, W. Lin, Y. Gal, and A. Srivastava. Fast and scalable bayesian deep learning by weight-perturbation in adam. In *International Conference on Machine Learning*, pages 2611–2620. PMLR, 2018.

[20] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[21] B. Yang, W. Luo, and R. Urtasun. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7652–7660, 2018.

[22] Y. Yan, Y. Mao, and B. Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10), 2018. ISSN 1424-8220. doi:10.3390/s18103337. URL https://www.mdpi.com/1424-8220/18/10/3337.

[23] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom. Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[24] C. R. Qi, O. Litany, K. He, and L. J. Guibas. Deep hough voting for 3d object detection in point clouds. In *proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9277–9286, 2019.

[25] D. Hall, F. Dayoub, J. Skinner, H. Zhang, D. Miller, P. Corke, G. Carneiro, A. Angelova, and N. Sünderhauf. Probabilistic object detection: Definition and evaluation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1031–1040, 2020.

[26] D. Feng, L. Rosenbaum, C. Glaeser, F. Timm, and K. Dietmayer. Can we trust you? on calibration of a probabilistic object detector for autonomous driving. *arXiv preprint arXiv:1909.12358*, 2019.

[27] D. Feng, L. Rosenbaum, F. Timm, and K. Dietmayer. Leveraging heteroscedastic aleatoric uncertainties for robust real-time lidar 3d object detection. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1280–1287. IEEE, 2019.

[28] Z. Wang, D. Feng, Y. Zhou, L. Rosenbaum, F. Timm, K. Dietmayer, M. Tomizuka, and W. Zhan. Inferring spatial uncertainty in object detection. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5792–5799, 2020. doi:10.1109/IROS45743.2020.9340798.

[29] A. Kendall, V. Badrinarayanan, and R. Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*, 2015.

[30] V. Kuleshov, N. Fenner, and S. Ermon. Accurate uncertainties for deep learning using calibrated regression. In *International conference on machine learning*, pages 2796–2804. PMLR, 2018.

[31] M. P. Naeini, G. Cooper, and M. Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[32] G. Yarin. Uncertainty in deep learning. *University of Cambridge, Cambridge*, 2016.

[33] J. Martens and R. Grosse. Optimizing neural networks with kronecker-factored approximate curvature. In *International conference on machine learning*, pages 2408–2417. PMLR, 2015.

[34] F. Kunstner, P. Hennig, and L. Balles. Limitations of the empirical fisher approximation for natural gradient descent. *Advances in neural information processing systems*, 32, 2019.

[35] J. K. Kruschke. Bayesian data analysis. *Wiley Interdisciplinary Reviews: Cognitive Science*, 1 (5):658–676, 2010.