A Review and Comparative Study on Probabilistic Object Detection in Autonomous Driving

Di Feng[†1], Ali Harakeh[†2], Steven Waslander[2], Klaus Dietmayer[1]

***Abstract*—Capturing uncertainty in object detection is indispensable for safe autonomous driving. In recent years, deep learning has become the de-facto approach for object detection, and many probabilistic object detectors have been proposed. However, there is no summary on uncertainty estimation in deep object detection, and existing methods are not only built with different network architectures and uncertainty estimation methods, but also evaluated on different datasets with a wide range of evaluation metrics. As a result, a comparison among methods remains challenging, as does the selection of a model that best suits a particular application. This paper aims to alleviate this problem by providing a review and comparative study on existing probabilistic object detection methods for autonomous driving applications. First, we provide an overview of generic uncertainty estimation in deep learning, and then systematically survey existing methods and evaluation metrics for probabilistic object detection. Next, we present a strict comparative study for probabilistic object detection based on an image detector and three public autonomous driving datasets. Finally, we present a discussion of the remaining challenges and future works. Code has been made available at https://github.com/asharakeh/pod_compare.git.**

***Keywords*—Uncertainty estimation, object detection, deep learning, autonomous driving**

I. Introduction

Capturing perceptual uncertainties is indispensable for safe autonomous driving. Consider a self-driving car operating *in snowy days*, when on-board sensors can be compromised by snow; *during the night-time*, when the image quality of RGB cameras is diminished; or *on an unfamiliar street*, where we encounter a motorized-tricycle, which can be often seen in Asian cities but are exceedingly rare in Western Europe. In these complex, unstructured driving environments, the perception module may make predictions with varied errors and increased failure rates. Determining reliable perceptual uncertainties, which reflect perception inaccuracy or sensor noises, could provide valuable information to introspect the perception performance, and help an autonomous car react accordingly. Further, cognitive psychologists have found that humans are good intuitive statisticians, and have a frequentist sense of uncertainties [1]. Therefore, reliable perceptual uncertainties could help humans better interpret the intention of autonomous cars, and enhance the development of trust in this rapidly evolving technology. As the machine learning methods (especially deep learning) have been widely applied to safety-critical computer vision problems [2], efforts to improve a network's self-assessment ability, reliability and interpretability with uncertainty estimation are steadily increasing [3], [4].

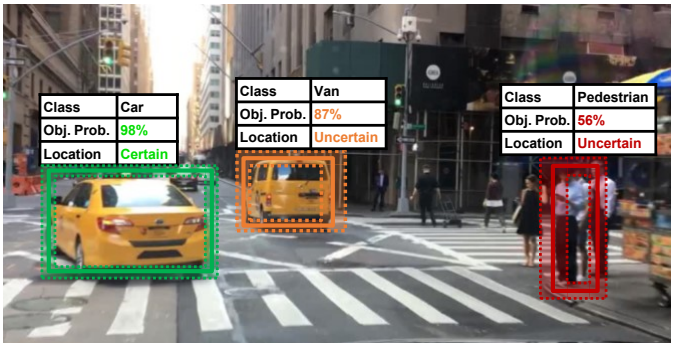


Fig. 1: A *conceptual* illustration of probabilistic object detection in an urban driving scenario. Each object is classified with a classification probability, and its bounding box is predicted with a confidence interval. The RGB camera image is from the BDD100k dataset [17].

In this paper, we focus on object detection, one of the most important perception problems in autonomous driving. An object detector is targeted to jointly classify and localize relevant traffic participants from on-board sensor data (e.g. RGB camera images, LiDAR and Radar points) in a frame-by-frame manner, as shown conceptually in Fig 1. In recent years, deep learning has become the de-facto approach in object detection, and many methods of modelling uncertainties in deep neural networks have been proposed [5]–[16]. However, to our knowledge, there is no work that provides a summary on uncertainty estimation in deep object detection, making it difficult for researchers to enter this field. Besides, existing probabilistic object detection models are often built with different network architectures, different uncertainty modelling approaches, and different sensing modalities. They are also tested on different datasets with a wide range of evaluation metrics. As a result, a comparison among methods remains challenging, as does the selection of the model that best suits a particular application.

Contributions

In this work, we systematically review the uncertainty estimation approaches that have been applied to object detection with deep learning approaches, and conduct a comparative

[†] Di Feng and Ali Harakeh contributed equally to this work. Listing order is random.

[1] Institute of Measurement, Control and Microtechnology, Ulm University, 89081 Ulm, Germany.

[2] Toronto Robotics and Artificial Intelligence Laboratory, University of Toronto Institute for Aerospace Studies (UTIAS), Toronto, Canada.

Corresponding author: di.feng@uni-ulm.de

study on probabilistic object detectors for autonomous driving applications.

First, we provide an overview of generic uncertainty estimation in deep learning (Sec. II). We summarize four practical uncertainty estimation methods, discuss the types of uncertainty that can be modelled in perception problems, and introduce common metrics to quantify uncertainties. Next, we survey existing methods and the evaluation metrics designed specifically for probabilistic object detection (Sec. III). We then present a strict comparative study for probabilistic object detectors (Sec. IV). Based on the one-stage 2D image detector RetinaNet [18], we benchmark several practical uncertainty estimation methods on three public autonomous driving datasets, namely, the KITTI object detection benchmark [19], the BDD100k Diverse Driving Video Database [17], and the Lyft Perception Dataset [20]. In addition, we compare how uncertainties behave in RGB camera images and LiDAR point clouds, as camera and LiDAR sensors have very different sensing properties and observation noise. Finally, we present a discussion of the remaining challenges in probabilistic object detection (Sec. VI).

Open source code for benchmarking probabilistic object detection is made available at: https://github.com/asharakeh/pod_compare.git.

II. Uncertainty Estimation in Deep Learning

This section presents background knowledge of general uncertainty estimation in deep learning. First, Sec. II-A defines the notation and problem formulation for predictive probability estimation in deep learning. Then, Sec. II-B introduces Bayesian Neural Networks (BNNs), which provide a natural interpretation of uncertainty estimation in deep learning. Based on the BNN framework, Sec. II-C decomposes predictive uncertainty into **epistemic** and **aleatoric** uncertainty, which will be widely used in this paper. Afterwards, Sec. II-D introduces four practical methods for uncertainty estimation in the literature, namely, Monte-Carlo Dropout, Deep Ensembles, Direct Modelling, and Error Propagation. Finally, Sec. II-E summarizes common evaluation metrics and benchmarks used in uncertainty estimation.

A. Notation and Problem Formulation

Let us denote a labelled training dataset of N data pairs as $\mathcal{D} = \{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^{N}$, where $\mathbf{x}_n$ is a data sample which represents high-dimensional sensor input in the domain $\mathcal{X}$, and $\mathbf{y}_n$ its corresponding target value (or target label in other words) in the domain $\mathcal{Y}$. A data pair $\mathbf{x}_n$, $\mathbf{y}_n$ is assumed to be an i.i.d. realization of jointly distributed random variables X, Y, with X being the input variable, and Y the target variable. For brevity, the subscript n is ignored in the following sections unless mentioned otherwise. In classification, a target label can be one of the C classes $\mathcal{Y} = \{1, 2, ..., C\}$. Therefore, it can be denoted as a positive integer $y \in \{1, \dots, C\}$. It can also be a binary vector $\mathbf{y} \in \{0, 1\}^C : \sum_{i=1,...C} y_i = 1$ for the one-hot encoding. In regression, a target value is usually a continuous vector with D dimensions denoted by $\mathbf{y} \in \mathbb{R}^D$, or a real value denoted by $y \in \mathbb{R}$ in an one-dimensional regression problem. In supervised learning, we aim to learn a model $f^{\mathbf{W}} : \mathcal{X} \rightarrow \mathcal{Y}$ parametrized with weights $\mathbf{W} \in \mathcal{W}$ from the training dataset $\mathcal{D}$, where $\mathcal{W}$ is the domain of weights. The model maps an input sample $\mathbf{x}$ to its corresponding output target value $\mathbf{y}$ via the model output prediction $\hat{\mathbf{y}}$. Given a test data sample during the test time, often denoted by the superscript of star $\mathbf{x}^*$, the model is expected to accurately estimate the unknown true target value $\mathbf{y}^*$ with its model prediction $\hat{\mathbf{y}}^*$. To simplify the notation, we will not distinguish between a data sample for training or testing unless necessary for clarity. Therefore, the superscript of star $\mathbf{x}^*$ is ignored.

Usually, deep neural networks are interpreted as point estimators with deterministic network weights $\mathbf{W}$, which provide deterministic output predictions denoted by $\hat{\mathbf{y}} = f(\mathbf{x}, \mathbf{W})$. When doing uncertainty estimation in supervised deep neural networks, we expand the network output definition to encompass a predictive probability distribution denoted by $p(\mathbf{y}|\mathbf{x}, \mathcal{D})$.

B. Bayesian Neural Networks

Bayesian Neural Networks (BNNs) were first introduced in the 1990s [21], [22], and have been a gold standard for probabilistic inference with neural networks since then. BNNs provide a natural interpretation of uncertainty estimation in deep learning, by inferring distributions over a network's weights $\mathbf{W}$. Given a data sample $\mathbf{x}$, BNNs produce the predictive distribution $p(\mathbf{y}|\mathbf{x}, \mathcal{D})$ by integrating over all values of network weights:

$$p(\mathbf{y}|\mathbf{x}, \mathcal{D}) = \int p(\mathbf{y}|\mathbf{x}, \mathbf{W}) p(\mathbf{W}|\mathcal{D}) \mathrm{d}\mathbf{W}. \tag{1}$$

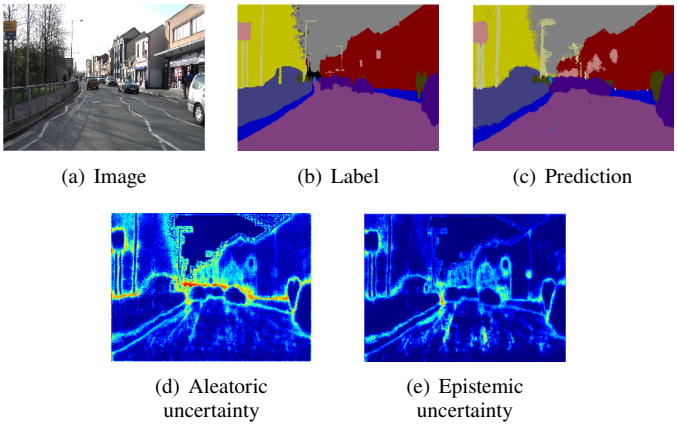
In this equation, $p(\mathbf{y}|\mathbf{x}, \mathbf{W})$ represents the observation likelihood, and $p(\mathbf{W}|\mathcal{D})$ represents the weight posterior distribution over the dataset. Modeling the observation likelihood is usually straight-forward and can be done using direct modeling, which we will introduce in Sec. II-D3. However, analytically calculating the posterior distribution is intractable due to its high dimensionality and multi-modality (often in a space of millions of weight parameters), as well as the non-linear activation function between consecutive layers of deep neural networks [23]. Multiple techniques for generating approximate solutions of the posterior distribution in Equation (1) have been proposed in the literature, including Variational Inference (VI) [24]–[26], Markov Chain Monte Carlo (MCMC) [27]–[29], Stochastic Gradient Descent (SGD) Approximation [30]–[32], and Laplace Approximation [33]. In practice, approximation techniques in BNNs are highly sensitive to hyperparameters, and hard to scale to large datasets and network architectures, and how to select an appropriate model prior and an approximate inference method remains an open question, as discussed in [32]. Recently, Gal *et al.* [34] proposed a practical method called Monte-Carlo Dropout (MC-Dropout) to perform inference in BNNs; we discuss the details of MC-Dropout in Sec. II-D1.

C. Epistemic and Aleatoric uncertainty

Predictive uncertainty in deep neural networks can be decomposed into *epistemic* uncertainty and *aleatoric* uncertainty [57]. *Epistemic*, or model uncertainty, indicates how

TABLE I: The applications of uncertainty estimation in autonomous driving and example references

Problem	Example Reference	Year	Unc. Type	Dataset
Camera pose estimate	Kendall *et al*. [35]	2016	E	Cambridge Landmarks [36]
Semantic segmentation	Kendell *et al*. [37]	2017	E	CamVid [38], SUN RGB-D [39], Pascal-VOC [40]
Image annotation	Mackowiak *et al*. [41]	2018	E	Cityscapes [42]
Optical flow	Ilg *et al*. [43]	2018	E+A	KITTI [19], MPI-Sintel Flow [44]
Learning by demonstration of driving control	Choi *et al*. [45]	2018	A	US Highway 101 [46]
End-to-end driving	Michelmore *et al*. [47]	2019	E	CARLA Simulation [48]
Trajectory prediction	Hu *et al*. [49]	2020	A	Self-recorded data
Depth completion	Walz *et al*. [50]	2020	A	Gated2Depth [51]
Visual tracking	Danelljan *et al*. [52]	2020	A	OTB100 [53], UAV123 [54], NFS [55] and four others



(a) Image (b) Label (c) Prediction

(d) Aleatoric uncertainty (e) Epistemic uncertainty

Fig. 2: An illustration of epistemic and aleatoric uncertainty in semantic segmentation on the CamVid dataset [38]. The epistemic uncertainty is measured by the Mutual Information, and the aleatoric uncertainty by the Shannon Entropy. Uncertainty estimation networks are adapted from Postel *et al*. [56]. The different behaviours of epistemic vs aleatoric uncertainty can be observed in (d) and (e). For example, the aleatoric uncertainty is more related to the object boundary, whereas the epistemic uncertainty is affected by the prediction inaccuracy (the front car is badly-segmented, and depict high epistemic uncertainty in (e)).

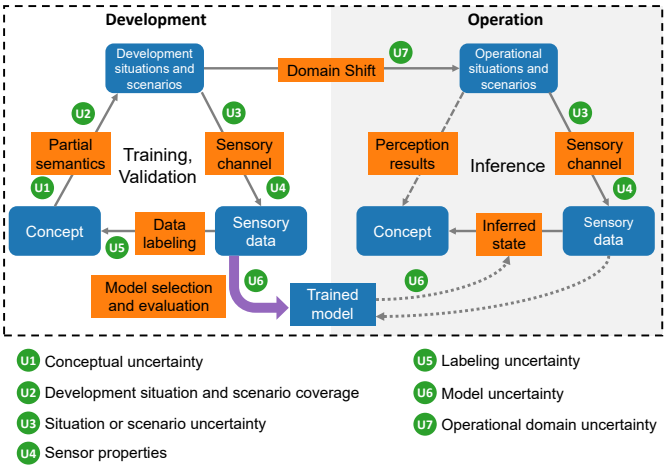


Fig. 3: Uncertainty categorization according to the procedure of developing and deploying a perception system. Seven distinct sources of uncertainty are defined. Adapted from [58].

certain a model is in using its parameters $\mathbf{W}$ to describe an observed dataset, and can be expressed by $p(\mathbf{W}|\mathcal{D})$ in Eq. 1. For instance, detecting an unknown object which is different from the training dataset is expected to show high epistemic uncertainty. *Aleatoric*, or data uncertainty, reflects observation noise inherent in sensor measurements of the environment, and can be modeled by $p(\mathbf{y}|\mathbf{x}, \mathbf{W})$ in Eq. 1. For example, detecting a distant object with only sparse LiDAR reflections, or using RGB cameras during the night drive should produce high aleatoric uncertainty. Capturing both types of uncertainty in a perception system is crucial for safe autonomous driving, as epistemic uncertainty displays the capability of a model and could be useful for out-of-data detection, while aleatoric uncertainty reflects sensor limitations in changing environments.

In addition to object detection, which we will explicitly introduce in Sec. III, epistemic and aleatoric uncertainty have been estimated in different components of autonomous driving such as semantic segmentation [37], [56], [59](e.g. Fig. 2), optical flow [43], [60], depth estimation [50], [57], [61], [62], visual odometry [35], [63], [64], image annotation [41], [65]–[67], visual tracking [52], trajectory prediction [49], [68]–[70] and end-to-end perception [47], [71], [72]. Tab. I lists several uncertainty estimation applications in autonomous driving. Note that although epistemic and aleatoric uncertainty are defined in the context of Bayesian Neural Networks, the two types of uncertainty can be captured by non-Bayesian methods such as Deep Ensembles and Direct Modelling which will be

introduced in Sec. II-D. Furthermore, both types of uncertainty are not mutually exclusive [57]; explicitly modeling only one may result in a predictive probability that reflects the properties of the other type of uncertainty as well.

Another Uncertainty Categorization Method: Beyond epistemic and aleatoric uncertainty, Czarnecki *et al*. [58] define seven sources of perceptual uncertainty to be considered for the development and operation of a perception system (such as object detection, semantic segmentation etc.), as shown by Fig. 3. During development, there exists uncertainty when defining the perception problems (U1), collecting and annotating data with selected scenarios (U2-U5), as well as training the model (U6). During operation, an additional type of uncertainty arises due to distribution shift between the training and testing data splits (U7).

D. Practical Methods for Uncertainty Estimation

In the following, we summarize four practical methods for predictive uncertainty estimation in deep learning: MC-Dropout, Deep Ensembles, Direct Modelling, and Error Propagation. MC-dropout and Deep Ensembles are used to model epistemic uncertainty, while Direct Modeling is used to capture aleatoric uncertainty. Error propagation can be used to capture either epistemic or aleatoric uncertainty, depending on its focus.

1) Monte-Carlo Dropout: Monte-Carlo Dropout (MC-Dropout), proposed by Gal *et al*. [34], links dropout-based neural network training to Variational Inference (VI) in Bayesian Neural Networks (BNNs), as introduced in Sec. II-B. Gal *et al*.show that training a network with Stochastic Gradient Descent (SGD) while dropout is enabled is equivalent to optimizing a posterior distribution that approximates Equation 1. The method is later extended in [73] to find optimal dropout rate by treating the dropout probability as a hyperparameter. During test time, samples from the approximate posterior distribution are generated by running the network multiple times with dropout enabled. Let T be the total number of feed-forward passes with dropout, and $\mathbf{W}_t$ be a sample of network weights after dropout. The predictive probability can be approximated from T generated samples as:

$$p(\mathbf{y}|\mathbf{x},\mathcal{D}) \approx \frac{1}{T}\sum_{t=1}^{T} p(\mathbf{y}|\mathbf{x},\mathbf{W}_t). \tag{2}$$

For classification, Eq. 2 corresponds to averaging the T number of predicted classification probabilities, such as softmax scores. For regression, this equation can be viewed as a mixture of distribution with T equally-weighted components. In this case, the sample mean and variance can be used to describe the predictive probability distribution. In general, MC-Dropout provides a practical way to perform approximate inference in Bayesian Neural Netowrks (BNNs), and is scalable to large datasets and network architectures such as in image classification [65] or semantic segmentation [61], [74]. However, MC-Dropout requires multiple stochastic runs during test time, usually 10 to 50 runs as shown in [34]. Therefore, estimating uncertainty with the MC-dropout is still infeasible for real-time critical systems due to its high computational cost.

2) Deep Ensembles: Deep Ensembles, proposed by Lakshminarayanan *et al*. [75], estimate predictive probability using an ensemble of networks where the output from each network is treated as independent samples from a mixture model. Each network in the ensemble follows the same architecture, but is trained with randomly shuffled training data using a different initialization of its parameters. Let M be the number of networks in an ensemble, $\{\mathbf{W}_m\}_{m=1}^{M}$ be their weights, and $p(\mathbf{y}|\mathbf{x},\mathbf{W}_m)$ be a prediction from the m-th network. The predictive probability is approximated by an uniformly-weighted mixture model with M components, given by:

$$p(\mathbf{y}|\mathbf{x},\mathcal{D}) \approx \frac{1}{M}\sum_{m=1}^{M} p(\mathbf{y}|\mathbf{x},\mathbf{W}_m). \tag{3}$$

The equation is similar to Eq. 2, as MC-dropout can also be interpreted as an ensemble of networks [75]. In practice, an ensemble of 5 networks has been shown to be sufficient to approximate predictive probabilities [75], [76]. Although easy to implement, the computation and memory costs of deep ensembles scale linearly with the number of networks both during training and inference, limiting its applicability for large network architectures, as discussed in [61].

3) Direct Modeling: Direct Modeling assumes a certain probability distribution over the network outputs, and uses the network output layers to directly predict parameters for such a distribution. Unlike Bayesian Neural Networks, which perform marginalization over weights $\mathbf{W}$ (Eq. 1), Direct Modeling uses point estimates of these weights to generate the predictive probability distribution $p(\mathbf{y}|\mathbf{x},\mathcal{D})$ through $p(\mathbf{y}|\mathbf{x},\mathbf{W})$.

The most common way to estimate the classification probability of the class c is by the softmax score $p(y=c|\mathbf{x},\mathbf{W}) = \hat{s}_c$, which is equivalent to a multinomial mass function. As for the regression probability, Gaussian distributions [57], [60] or Gaussian Mixture Models (GMM) [14], [45] are often used. For instance, we often assume that the target value y in a one-dimensional regression problem is Gaussian distributed, given by: $p(y|\mathbf{x},\mathbf{W}) = \mathcal{N}\big(y|\hat{\mu}(\mathbf{x},\mathbf{W}),\hat{\sigma}^2(\mathbf{x},\mathbf{W})\big)$. Here, the mean, $\hat{\mu}(\mathbf{x},\mathbf{W})$, is the network standard prediction i.e. $\hat{\mu}(\mathbf{x},\mathbf{W}) = f(\mathbf{x},\mathbf{W})$. The variance, $\hat{\sigma}^2(\mathbf{x},\mathbf{W})$, is also predicted by the network with an additional output layer.

Optimizing a probabilistic network using Direct Modeling can be viewed as a maximum likelihood problem, where $\mathbf{W}$ are optimized to maximize the observation likelihood of training data. In practice, we minimize the negative log likelihood $L(\mathbf{x},\mathbf{W}) = -\log\big(p(\mathbf{y}|\mathbf{x},\mathbf{W})\big)$ for a training data pair $\mathbf{x},\mathbf{y}$. When using the softmax function for classification, $L(\mathbf{x},\mathbf{W})$ is widely known as the cross-entropy loss. When using the Gaussian distribution for regression, the negative log likelihood can be written as:

$$L(\mathbf{x},\mathbf{W}) = \frac{\big(y-\hat{\mu}(\mathbf{x},\mathbf{W})\big)^2}{2\hat{\sigma}^2(\mathbf{x},\mathbf{W})} + \frac{\log\hat{\sigma}^2(\mathbf{x},\mathbf{W})}{2}. \tag{4}$$

The negative log likelihood in Equation 4 can be viewed as the standard L_2 loss (i.e. $\big(y-\hat{\mu}(\mathbf{x},\mathbf{W})\big)^2$) being weighted by the inverse of the predicted variance $\hat{\sigma}^2(\mathbf{x},\mathbf{W})$, and regularized with the $\log\hat{\sigma}^2(\mathbf{x},\mathbf{W})$ term.

Instead of using the standard softmax with the cross-entropy loss for learning categorical distributions, Kendall *et al*. [57]

combine the softmax function with a Gaussian distribution to estimate classification uncertainty. Classification uncertainty is learnt by assuming that each element in the softmax logit vector is independently Gaussian distributed, with its mean and variance directly predicted by the network output layers. Let a softmax logit regression variable for the class c be l_c, its distribution is given in the form: $p(l_c|\mathbf{x},\mathbf{W}) = \mathcal{N}\big(l_c|\hat{\mu}_c(\mathbf{x},\mathbf{W}),\hat{\sigma}_c^2(\mathbf{x},\mathbf{W})\big)$, where $\hat{\mu}_c(\mathbf{x},\mathbf{W}),\hat{\sigma}_c^2(\mathbf{x},\mathbf{W})$ refer to the predicted mean and variance of the softmax logit. Training this softmax logit is different from the normal regression variable shown in Eq. 4, as there are only ground truth for object categories. In this regard, Kendall *et al*. [57] propose to sample the softmax logit based on the predicted Gaussian distribution via the re-parametrization trick, and then transform the sampled softmax logit into the softmax score to calculate a standard classification loss, such as the cross-entropy loss or the focal loss [18].

In addition to directly predicting the output probability distributions, several works propose to estimate their higher-order conjugate priors. For example, Malinin *et al*. [77] and Sensoy *et al*. [23] predict the Dirichlet prior for the softmax function in classification. Amini *et al*. [78] place a Gaussian prior on the mean and an Inverse-Gamma prior on the variance for the probability distribution in regression, which is assumed to be Gaussian distributed. Finally, Gustafsson *et al*. [79] use a deep neural network to directly predict the conditional target density of an energy-based model.

Since Direct Modeling requires one inference run, it brings almost no additional computation cost when used to capture uncertainty, but requires modifications to the network's output layers and training with a new loss function. Direct Modeling has also been shown to produce mis-calibrated probabilities for both classification [34], [80] and regression [81] tasks.

4) Error Propagation: Error Propagation approximates variances (or uncertainty) in each activation layer, and then propagates variances through the whole network from input layers to output layers. For example, Postels *et al*. [56] view dropout and batch normalization as a noise-injection procedure to learn uncertainty during training and propose to approximate dropout error as a covariance matrix in the noisy layers. They then propagate the error through the downstream activation layers in closed-form. In this way, the method replaces the time-consuming MC-dropout [34] using a single inference, significantly reducing the computation time. Similarly, Gast *et al*. [60] convert a standard activation layer such as ReLu into an uncertainty propagation layer by matching its first and second-order central moments. Due to the computational efficiency at inference and limited modifications required for training, error propagation appeals to practitioners with real-world applications.

E. Evaluation and Benchmarking

1) Shannon Entropy : Shannon entropy (SE) is a common evaluation metric to estimate the quality of predictive uncertainty for classification tasks [34]. For a class label y of C classes, the SE, $\mathcal{H}$, is measured by:

$$\mathcal{H}(y|\mathbf{x},\mathcal{D}) = -\sum_{c=1}^{C} p(y=c|\mathbf{x},\mathcal{D}) \log\big(p(y=c|\mathbf{x},\mathcal{D})\big). \quad (5)$$

SE reaches a minimum value when the network is certain in its prediction, i.e. $p(y=c|\mathbf{x},\mathcal{D}) = 0$ or 1, and maximum when the prediction follows an uniform distribution, i.e. $p(y=c|\mathbf{x},\mathcal{D}) = \frac{1}{C}$.

2) Mutual Information: Another common evaluation metric for measuring the quality of uncertainty in classification tasks is Mutual Information (MI). A practical MI formulation was introduced in [34], which measures the information gain of the predictive probability, when introducing the posterior distribution of model parameters $\mathbf{W}$:

$$\mathcal{I}(y,\mathbf{W}|\mathbf{x},\mathcal{D}) = \mathcal{H}(y|\mathbf{x},\mathcal{D}) - \mathbb{E}_{p(\mathbf{W}|\mathcal{D})}\big[\mathcal{H}(y|\mathbf{x},\mathbf{W})\big], \quad (6)$$

where $\mathbb{E}[\cdot]$ is the expectation operation, and $\mathcal{H}(y|\mathbf{x},\mathbf{W})$ is the conditional Shannon Entropy with respect to $\mathbf{W}$, given by:

$$\mathcal{H}(y|\mathbf{x},\mathbf{W}) = -\sum_{c=1}^{C} p(y=c|\mathbf{x},\mathbf{W}) \log\big(p(y=c|\mathbf{x},\mathbf{W})\big). \quad (7)$$

The first term on the right-hand side of Eq. 6 is calculated using Eq. 5. The expectation in the second term is approximated by averaging the network's predictions with the network's weight samples (e.g. sampled from the MC-Dropout [34] or the Deep Ensembles [75] approaches). Different from SE which directly measures the predictive uncertainty, MI captures the variations within a model brought by the network's weights $\mathbf{W}$. Therefore, it reflects the model uncertainty. An MI score ranges between $[0,1]$, with a larger value indicating higher uncertainty.

3) Calibration Plot: A calibration plot is a common tool to evaluate the uncertainty calibration quality of a probabilistic model. From the frequentist perspective, a well-calibrated probabilistic model should predict uncertainty that match the natural frequency of correct predictions. For example, if a deep network makes 100 predictions with 0.8 probability (or 80% confidence interval in the regression task), then 80% of those predictions should be correct (or fall into such a confidence interval). A calibration plot reflects this information by drawing a curve with the predicted probability on the horizontal axis and its corresponding empirical probability on the vertical axis, as shown in Fig. 4. A well-calibrated deep network produces a diagonal line. However, Guo *et al*. [80] and Kuleshov *et al*. [81] have empirically identified the over- or under-confident uncertainty in many classification and regression problems, respectively. They propose uncertainty recalibration techniques such as isotonic regression and temperature scaling to improve posterior distribution calibration. Note that calibration plotting provides a natural representation of the reliability of uncertainty estimation for a whole dataset, but cannot represent the uncertainty performance for each data point individually.

To draw a calibration plot for an evaluation dataset with the size N_{eval}, the network needs to run probabilistic predictions for all data samples. The predicted probabilities from the network, denoted by p, are partitioned into T intervals (or

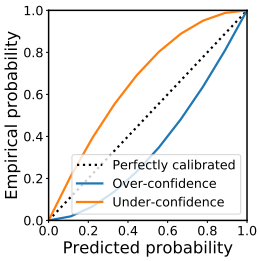


Fig. 4: A calibration plot. The horizontal axis represents the predictive probability, and the vertical axis the empirical probability. A miscalibrated network can make over-confident or under-confident predictions, as indicated in blue and orange curves, respectively.

quantiles), i.e. $0 < p_1 < ... < p_t < ... < 1$. They correspond to the values in the horizontal axis of Fig. 4. In each interval, the normalized empirical frequency denoted by $\hat{p}_t$ is calculated by comparing predictions with ground truths. An empirical frequency corresponds to a value in the vertical axis of Fig. 4.

From the calibration plot we can derive several evaluation metrics. For example, Expected Calibration Error (ECE) [80] measures the weighted absolute error between the actual calibration curve and the optimal calibration curve (i.e. the diagonal line). A ECE score is given by:

$$\text{ECE} = \sum_{t=1}^{T} \frac{N_t}{N_{eval}} |p_t - \hat{p}_t|, \tag{8}$$

where p_t is the predicted probability from a network, $\hat{p}_t$ the empirical frequency, and N_t the number of samples in the t-th interval, $\sum_{t=1}^{T} N_t = N_{eval}$. An ECE score ranges between $[0, 1]$ with smaller value indicating better uncertainty calibration performance. Besides ECE, Average Calibration Error (ACE) and Maximum Calibration Error are also common metrics. ACE calculates the absolute error averaged over all intervals equally, while the Maximum Calibration Error finds the maximum absolute error in all intervals. Finally, Kumar *et al*. [82] proposed the Marginal Calibration Error (MCE) metric, as an extension of ACE. While ACE considers the uncertainty calibration quality only on an object ground truth category, MCE measures the uncertainty of a classifier's predicted distribution over all categories of interest.

4) Negative Log Likelihood (NLL): NLL is a standard metric to measure probabilistic models in a test dataset with N_{test} data points. NLL is calculated by:

$$\text{NLL} = -\sum_{n=1}^{N_{test}} \log\big(p(\mathbf{y}_n|\mathbf{x}_n, \mathcal{D})\big), \tag{9}$$

where $\mathbf{x}_n$ is a test data point, and $\mathbf{y}_n$ its corresponding ground truth label. NLL ranges in $(-\infty, +\infty)$. A lower NLL score indicates a better fitting predictive distribution for that specific ground truth label. Therefore, NLL indirectly reflects the uncertainty calibration quality.

5) Brier Score: A Brier Score (BS) is another common metric used to measure the accuracy of probabilistic predictions specifically in classification. It is calculated by the squared error between a predictive probability from the network and its one-hot encoded ground truth label. Given N_{test} number of test data samples, for a classification problem with C number of classes, a Brier Score is given by:

$$\text{BS} = \frac{1}{N_{test}} \sum_{n=1}^{N_{test}} \sum_{c=1}^{C} (\hat{s}_{n,c} - y_{n,c})^2, \tag{10}$$

where $\hat{s}_{n,c}$ is a predicted classification score for the c-th class, such as the softmax score, and $y_{n,c}$ is ground truth label ("1" if the ground truth class is c and "0" otherwise). Brier scores range between 0 and 1, with smaller values indicating better uncertainty estimation.

6) Error Curve: This method proposed by Ilg *et al*. [43] measures how predictive uncertainty matches true prediction errors. It assumes that a well-estimated predictive uncertainty should correlate with the true error, and by gradually removing the predictions with the highest uncertainty, the average errors in the rest of the predictions will decrease. To draw an error curve, a model makes predictions the uncertainty estimates through a test dataset. Those predictions are ranked according to their uncertainties, such as the Shannon Entropy or Mutual Information scores in classification, and predicted variances in regression. The error curve reports the percentage of removed testing samples according to their uncertainty ranking on the horizontal axis, and the averaged errors in the rest of the predictions on the vertical axis (such as the cross entropy errors in classification, and mean squared errors in regression).

7) Total Variance (TV): The metric proposed by Feng *et al*. [7] is designed to measure the dispersion of a probability distribution in regression tasks, by calculating the trace of its covariance matrix, i.e. summing up all variances in the diagonal line of the covariance matrix. A TV score ranges within $[0, +\infty)$, with a larger value indicating higher uncertainty. Note that TV only quantifies the variance in each regression variable, and ignores the correlations among regression variables.

Benchmarking Uncertainty Estimation Methods: Previous works have evaluated and compared existing uncertainty estimation methods mainly in open-world scenarios. In this setting, the test data distribution is different from the training dataset (also known as *domain shift*), or there exist objects which a perception module has never seen before (also known as *out-of-distribution*). This setting is of great interest from the practical perspective of applications such as autonomous driving. Snoeck *et al*. [76] conduct a large-scale study to benchmark classification uncertainty estimation, including the image classification using MNIST, ImageNet [83] and CIFAR [84] datasets, as well the text and ad-click classification. They compare softmax output, MC-dropout [34], Deep Ensembles [75], temperature scaling (as the post-hoc calibration tool) [80], as well as several Bayesian inference approaches under the domain shift setting. Filos *et al*. [85] benchmark uncertainty estimation methods in the medical image application (diabetic retinopathy diagnosis). Blum *et al*. [74] build the Fishyscapes dataset to benchmark uncertainty estimation methods in semantic segmentation for anomaly detection. The dataset is based on Cityscapes [42], which is a popular semantic segmentation dataset for urban autonomous driving. From the above-mentioned works, we have two observations: (1). There is a trade-off between uncertainty estimation quality and the model's accuracy, that is, a model with the best uncertainty estimates when faced with domain-shift or out-of-distribution data may have lower detection accuracy. (2). There is no conclusive evidence that one method is much better than the others. Uncertainty estimation in deep learning is still an

open question, especially in the domain shift context and in terms of scalability for large modern neural networks.

III. Probabilistic Object Detection

In this section, we provide a systematic summary of probabilistic object detection using deep learning approaches. We start with a background introduction of generic object detection (Sec. III-A), and then summarize each probabilistic object detection method in detail (Sec. III-B). Afterwards, we list several common evaluation metrics and discuss their properties (Sec. III-C). Finally, we briefly summarize sensing modalities and use cases for existing probabilistic object detection methods in Sec. III-D and Sec. III-C, respectively.

A. Background

Object detection is a multi-task problem that requires to jointly recognize objects with classification scores, and localize objects with tightly fitting bounding boxes. State-of-the-art object detectors using deep learning can follow one-stage, two-stage, or the recently-proposed sequence-to-sequence-mapping detection pipelines. In the one-stage object detection pipeline, a deep learning model is employed to directly map the input data to bounding boxes and classification scores in a single shot. Typical architectures include YOLO [86], SSD [87], and RetinaNet [18], to name a few. In contrast, the two-stage object detection pipeline extracts several class-agnostic object candidates called regions of interests (ROIs) or region proposals (RPs) in the first stage, and refines their class and bounding box attributes in the second stage. Those ROI can be extracted by non-deep learning approaches such as clustering [88] and voting [89], or by a neural network such as a Region Proposal Network (RPN). State-of-the-art two-stage image detectors are in general based on the Faster-RCNN architecture [90]. Usually, both one-stage and two-stage pipelines require a prior anchor grid and an output post-processing stage such as Non-Maximum Suppression (NMS) to suppress duplicate detection results. Recently, Tian *et al*. [91] propose an anchor-free detector by estimating objects at every pixel, and designing a center-ness loss to allow efficient training of such formulation. Carion *et al*. [92] take a more drastic approach to avoid prior anchor grid, by reformulating the object detection task as a sequence-to-sequence mapping problem, leading to Detection Transformers (DETR). DETR directly maps a sequence of query points into objects in the scene without the prior anchor grid. The method also avoids duplicate detections through training with the optimal positive-negative sample assignment, obviating the need of a post-processing step such as NMS.

Different from images which are mostly processed by the standard 2D CNN, there are several methods in object detection to process point clouds from LiDAR and Radar sensors, as well as pseudo point cloud from Stereo cameras. Those point clouds can be projected onto 2D feature maps, and processed by standard 2D CNNs [93]–[96]. They can also be discretized in voxel [97]–[101] or processed in the continuous vector space without voxelization [102]–[104]. For a more detailed summary we refer interested readers to [105] for object detection in 3D point clouds, and [106] for object detection in 2D images.

In general, state-of-the-art object detectors are not designed to capture reliable predictive uncertainty. They predict bounding box regression variables without any uncertainty estimation, and usually classify objects with softmax scores, which may not necessarily represent reliable classification uncertainties (as discussed in [34]). As a result, most object detectors are *deterministic*. They only predict *what* they have seen, but not *how uncertain* they are about it. In this regard, probabilistic object detectors are targeted to predict reliable uncertainties both in object classification and bounding box regression tasks.

B. Methodology

Probabilistic object detectors usually extend the network architectures described in the previous Section (Sec. III-A) to provide uncertainty estimates in their outputs, especially in bounding box predictions. Fig. 5 provides an overview of the general building blocks to perform probabilistic object detection, which mainly consists of a base network, a detection head, and a post-processing stage. Note that both one stage and two stage approaches can be extended to predict uncertainty in a similar manner, and thus we do not distinguish between the two for the remainder of this paper.

A base networks (Fig. 5, Left) maps sensor inputs to high-level feature maps, which will be further processed by a detection head. Probabilistic object detectors usually adapt base networks from well-studied deterministic object detectors to capture epistemic uncertainty using Deep Ensembles or MC-Dropout [7], [105], [107]–[109]. Typical base networks include VGG [110], ResNet [111], Inception [112], to name a few. *A detection head* (Fig. 5, Middle) takes the feature maps from the base network as inputs and predicts probabilistic detection outputs. Deep Ensembles [105], [108] and MC-Dropout [5], [6], [12], [16], [107]–[109] are usually used to model epistemic uncertainty in the detection head. On the other hand, the direct modelling approach, introduced in Sec. II, models aleatoric uncertainty by introducing additional output layers to the detection head for estimating the variance of output bounding boxes [7], [8], [10]–[16], [30], [105], [107], [109], [113]–[117] or the variance of output softmax logit vectors [11], [113]. Finally, *a post-processing stage* is used to suppress or merge redundant detection outputs as is common in deterministic object detectors. Tab. II summarizes the post-processing step in state-of-the-art probabilistic object detectors. The majority of methods employs the standard Non-Maximum Suppression (NMS), because it is a common component from deterministic object detectors, upon which probabilistic object detectors are built. Other methods [14], [15], [114] modify NMS to take into account spatial uncertainty when deciding which redundant object to remove, or replace NMS with Bayesian Inference [12] to include information from all redundant detection outputs. Finally, [5], [6], [113] take the benefits of redundant detection outputs to compute sample statistics of probability distributions, instead of directly-modelling the output probability and discarding redundant detections.

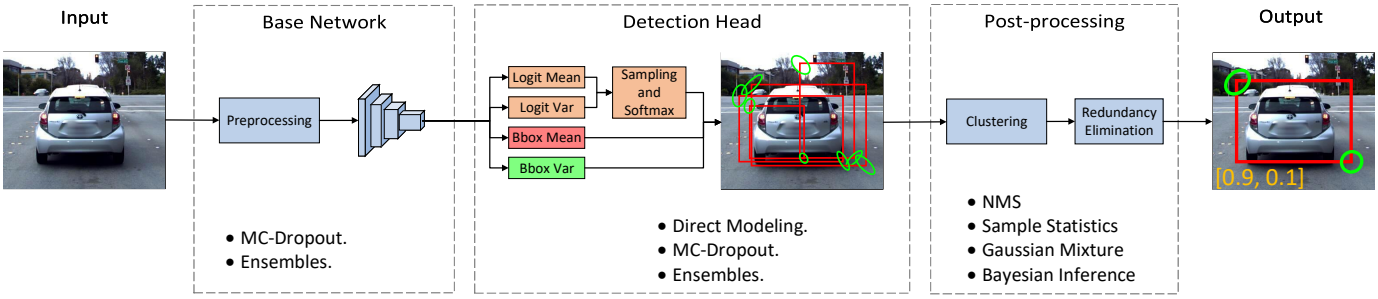


Fig. 5: Illustration of the key building blocks usually present in state-of-the-art probabilistic object detectors, including the base network, detection head and post-processing stages. A list of possible variants of each building block is also presented bellow the architectural diagram. The output detections on 2D images are visualized as category probabilities (**orange**), a bounding box mean (**red**), and the 95% confidence iso-contours of the bounding box corner covariance matrices (**green**).

The following section summarizes each state-of-the-art probabilistic object detector in detail. They are grouped based on what kinds of uncertainty are modelled, namely, epistemic uncertainty, aleatoric uncertainty, or both. Tab. II provides an overview of all methods we have reviewed.

1) Probabilistic Object Detectors with Epistemic Uncertainty: Several works model epistemic uncertainty by the MC-Dropout approach. Miller *et al*. [5] estimate epistemic uncertainty based on a SSD [87] network. The detection head is modified with dropout layers to generate samples during test time with dropout inferences. At the post-processing step, output samples from multiple stochastic MC-dropout runs are gathered, after performing the NMS operation on each stochastic run independently. Those redundant detections from multiple inferences are then clustered using spatial and/or semantic affinity. For every cluster, the probability distribution describing a single object is modeled using the sample mean and variance of the cluster members. The same architecture is used in [6] for studying the effect of various clustering techniques on the quality of epistemic uncertainty. It was concluded that simple clustering techniques such as Basic Sequential Algorithmic Scheme (BSAS) provides higher quality uncertainty values, when compared to more complicated clustering techniques such as the Hungarian algorithm.

The successive work by Miller *et al*. [108] avoid the clustering process when computing epistemic uncertainty. They build two detectors based on the Faster-RCNN [90] and SSD [87] architectures, each generates multiple samples at a certain anchor grid location by the MC-dropout inference. Those samples are then used to compute summary statistics at every anchor location using Eq. 2 or Eq. 3. Finally, redundant detections are merged via the standard NMS operation. Experimental results show that combining MC-Dropout inferences from different network architectures outperforms the MC-Dropout inferences from a single POD network, in terms of the epistemic uncertainty estimation.

Finally, Feng *et al*. [9] use MC-Dropout [34] and Deep Ensemble [75] methods to estimate epistemic uncertainty for the category classification only. The classification uncertainty is employed to train a 3D LIDAR object detector in an active learning framework. Dropout layers are used only in the detection head of the proposed 3D detection architecture, and the standard NMS operation is used at the post-processing step.

2) Probabilistic Object Detectors with Aleatoric Uncertainty: Most probabilistic object detectors use the direct-modeling approach introduced in Sec. II to estimate aleatoric uncertainty. These detectors are often built following four steps 1) selecting a deterministic object detector as the basic network, 2) assuming a certain probability distribution in its outputs, 3). using additional layers at the detection head to regress probability parameters, and 4) training the modified detector by incorporating uncertainty in the loss function. The direct-modelling approach has been used in various object detection architectures, including SSD [113], Faster-RCNN [8], [11], [13], FCOS [118], Point-RCNN [115], and PIXOR [10].

The softmax function is widely used to estimate classification probability, which corresponds to a multinomial mass function. Le *et al*. [113] and Feng *et al*. [11] additionally assume a Gaussian distribution in their softmax logit. As for the regression probability, a majority of works assume that each bounding box regression variable is independent and follows a simple probability distribution, such as a univariate Gaussian distribution, Laplace distribution, or a combination of both (Huber mass function). The regression variables usually include the bounding box centroid positions, extents (length, width, height), and orientations. Though this probability assumption is simple and straight-forward, it ignores correlations among regression variables, and may not fully reflect the complex uncertainties of bounding boxes, especially when objects are occluded. Instead, Pan *et al*. [115] transform the regression variables back to the eight-corner representation for 3D bounding boxes, and directly learn the corner uncertainty; Meyer *et al*. [14] place a mixture of Gaussians on each regression variable; Harakeh *et al*. [12] learn a multivariate Gaussian distribution with the full covariance matrix for regression variables. The recent work by He *et al*. [119] propose a more generic probability distribution which considers both correlations and multi-modal behaviours. They use a multivariate mixture of Gaussians to describe 2D bounding boxes, and show its superiority to univariate Gaussian, multi-variate Gaussian, and a univariate mixture of Gaussian.

Training probabilistic object detectors to predict aleatoric uncertainty is usually achieved by minimizing the Negative

Log Likelihood, resulting in the well-known cross entropy loss for classification, and the attenuated regression loss (Eq. 4) for bounding box regression with Gaussian distributions. In contrast, [13], [15] introduce a prior distribution related to the ground truth bounding box parameters, and minimize the Kullback-Leibler divergence between the predictive probability distribution and the prior distribution. In this way, the predictive uncertainty is regularized with the prior distribution, which helps stabilize training and improve detection performance [15], [120].

In addition to the standard Non-Maximum Suppression (NMS) operation, some works leverage aleatoric uncertainty at the post-processing step to merge duplicate detections. For instance, Meyer *et al*. [14] propose to group detections using mean-shift clustering over corners, and combine the bounding box uncertainty and classification scores in an uncertainty-aware NMS frawework. Similarly, Choi *et al*. [114] design a detection criterion that considers both regression and classification uncertainty. This criterion is used to rank detections while performing the standard NMS.

Instead of the direct-modelling method, a unique approach based on output redundancy is proposed in [113] to model aleatoric uncertainty. This output redundancy method replaces the standard NMS with spatial clustering, and uses the Intersection-Over-Union (IOU) metric as a measure for detection affinity. A single probability distribution describing each output detection can then be generated by computing the sample mean and variance of cluster members. Though the output redundancy method has been shown in [12] to produce lower quality uncertainty estimates than the direct-modelling method (measured with PDQ), it is time-efficient, and requires only a small modifications to the original deterministic detection network.

3) Probabilistic Object Detectors with both Aleatoric and Epistemic Uncertainties: Aleatoric and epistemic uncertainties are jointly estimated mainly by the direct-modelling approach together with the MC-dropout or Deep Ensembles approaches. Given the detection samples from T stochastic runs of dropout inference (or M models in an network ensemble), and assuming that bounding boxes are Gaussian distributed, the mean and variance of a bounding box regression variable can be computed by an uniformly-weighted Gaussian mixture model. Using the same notation from Sec. II-A, we have:

$$\hat{\mu}(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T}\hat{\mu}(\mathbf{x}, \mathbf{W}_t) \tag{11}$$

$$\hat{\sigma}^2(\mathbf{x}) = \hat{\sigma}_e^2(\mathbf{x}) + \hat{\sigma}_a^2(\mathbf{x}) \tag{12}$$

$$\hat{\sigma}_e^2(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T}\left(\hat{\mu}(\mathbf{x}, \mathbf{W}_t)\right)^2 - \left(\hat{\mu}(\mathbf{x})\right)^2 \tag{13}$$

$$\hat{\sigma}_a^2(\mathbf{x}) = \frac{1}{T}\sum_{t=1}^{T}\hat{\sigma}^2(\mathbf{x}, \mathbf{W_t}) \tag{14}$$

where $\hat{\mu}(\mathbf{x}, \mathbf{W}_t) = f(\mathbf{x}, \mathbf{W}_t)$ and $\hat{\sigma}^2(\mathbf{x}, \mathbf{W}_t)$ are the predicted mean and variance of the bounding box or classification logit vector outputs of sample $t \in [1 \dots T]$. Furthermore, $\hat{\sigma}_e^2(\mathbf{x})$ is the epistemic component of uncertainty, and is estimated as the sample variance using T output samples, and $\hat{\sigma}_a^2(\mathbf{x})$ is the aleatoric component of uncertainty, and can be computed as the average of T predicted variance estimates $\hat{\sigma}^2(\mathbf{x}, \mathbf{W}_t)$. Note that the final mean $\hat{\mu}(\boldsymbol{x})$ and variance $\hat{\sigma}^2(\boldsymbol{x})$ do not depend on the weights $\mathbf{W}$, because this variable is marginalized using averaging over T samples. For more details on the uniformly-weighted Gaussian mixture model in Eq. (12)-(14), we refer the reader to [75].

While all existing works use the output layers to directly model aleatoric uncertainty, they are different in epistemic uncertainty estimation. For example, Feng *et al*. [7] add dropout layers and perform multiple inferences only in the detection head of a Faster-RCNN object detector. Wirges *et al*. [16] follow the same network meta-architecture, and study the effect of adding dropout layers in either the base network or the detection head. Harakeh *et al*. [12] modify a 2D image detector based on RetinaNet [18], and incorporate MC-dropout in the detection head. Kraus *et al*. [107] extend a Yolov3 network, and add dropout inference after each convolutional layer in both the base network and the detection head.

As for the post-processing step, [7], [107], [121] use standard NMS. Harakeh *et al*. [12] replace it with Bayesian inference, which combines all information from redundant detections to generate the final output. Prior to the Bayesian inference step, a Gaussian mixture model is used to fuse epistemic and aleatoric uncertainty estimates per anchor. This method is shown to provide a large increase in uncertainty quality compared to the methods using standard NMS.

C. Evaluation Metrics

Based on the papers reviewed, there seems to be little agreement on how to evaluate probabilistic object detectors. Tab. II lists the evaluation metrics for each probabilistic object detection method in the literature, where it can be seen that each method uses different evaluation metrics, ranging from simple true positives and false positives [113] to complex metrics such as Probability-based Detection Quality (PDQ) [132]. The following section summarizes the common evaluation metrics, and discusses their properties when evaluating probabilistic object detection.

1) Precision, Recall and the F1-Score: In the context of object detection, only predictions above a pre-defined classification score threshold, δ_{cls}, are considered for evaluation. They are further divided into True Positives (TP) and False Positives (FP), based on their Intersection over Union (IOU) scores with ground truths. Detections above a certain pre-defined IOU threshold, δ_{iou}, are considered as True Positives. Redundant detections, those below the IOU threshold, are False Positives. False Negatives (FN) are further defined as the ground truth bounding boxes, which are either missed or unassociated with predictions due to low IOU scores. The counts of TP and FP were used in [113] to evaluate their proposed probabilistic object detector. TP and FP scores were also used to construct the Receiver Operating Characteristic (ROC) curves in [6], [116].

TABLE II: A summary of probabilistic object detectors

Reference	Year	Sensor†	Dataset	Unc. Type‡	Modeling Method	Post-processing	Evaluation Metrics
Feng *et al*. [7]	2018	L	KITTI [19]	E+A	MC-dropout, Direct Modeling	NMS	F-1 Score
Feng *et al*. [8]	2018	L	KITTI [19]	A	Direct Modeling	NMS	Average Precision
Le *et al*. [113]	2018	C	KITTI [19]	A	Direct Modeling, Output Redundancy	NMS, Output Statistics	True Positive/False Positive count
Miller *et al*. [5]	2018	C	COCO [122], SceneNet RGB-D [123], QUT Campus Dataset [124]	E	MC-dropout	NMS, Clustering, Sample Statistics	Precision and Recall, Absolute Open-set Error, F1-Score
Choi *et al*. [114]	2019	C	BDD [17], KITTI [19]	A	Direct Modeling	Uncertainty-aware NMS	Average Precision
Feng *et al*. [10]	2019	L	KITTI [19], NuScenes [125]	A	Direct Modeling	NMS	Average Precision, Expected Calibration Error
He *et al*. [13]	2019	C	COCO [122], Pascal-VOC [40]	A	Direct Modeling	NMS	Average Precision
Kraus *et al*. [107]	2019	C	EuroCity Persons [126]	E+A	MC-dropout, Direct modeling	NMS	Log Average Miss Rate
Meyer *et al*. [14]	2019	L	KITTI [19], ATG4D [96]	A	Direct modeling	Mean-shift clustering, Uncertainty-aware NMS	Average Precision
Meyer *et al*. [15]	2019	L	KITTI [19], ATG4D [96]	A	Direct modeling	Mean-shift clustering, Uncertainty-aware NMS	Average Precision
Miller *et al*. [6]	2019	C	COCO [122], VOC [40], Underwater Scenes	E	MC-dropout	NMS	Average Precision, Uncertainty Error, AUPR, AUROC
Miller *et al*. [108]	2019	C	COCO [122]	E	MC-Dropout, Deep Ensembles	Sample Statistics, NMS	PDQ
Wirges *et al*. [16]	2019	L	KITTI [19]	E+A	MC-dropout, Direct modeling	NMS	Average Precision
Lee *et al*. [118]	2020	C	COCO [122]	A	Direct modeling	NMS	Average Precision
Chen *et al*. [127]	2020	C	KITTI [19]	A	Direct modeling	Pairwise spatial constrain optimization	Average Precision
Dong *et al*. [117]	2020	R	Self-recorded data	A	Direct modeling	Soft NMS [128]	Average Precision
Feng *et al*. [11]	2020	L+C	KITTI [19], NuScenes [125], Bosch	A	Direct modeling	NMS	Average Precision
Harakeh *et al*. [12]	2020	C	COCO [122], VOC [40], BDD [17], KITTI [19]	E+A	MC-dropout, Direct Modeling	Bayesian Fusion	Average Precision, Uncertainty Error, PDQ
He *et al*. [119]	2020	C	COCO [122], VOC [40], CrowdHuman [129], VehicleOcclusion [130]	A	Direct modeling	Mixture of Gaussians, NMS	Average Precision, PDQ
Pan *et al*. [115]	2020	L	KITTI [19]	A	Direct modeling	NMS	Average Precision
Wang *et al*. [116]	2020	L	KITTI [19], Waymo [131]	A	Direct modeling	NMS	Average Precision, ROC curves, Jaccard-IOU

† L: LiDAR, C: RGB Camera, R: Radar ‡ A: Aleatoric uncertainty, E: Epistemic uncertainty

Based on TP, FP, and FN, the evaluation metrics Precision, Recall, and the F1-Score can be derived, given by:

$$\begin{aligned} \text{Precision} &= \frac{\text{TP}}{\text{TP}+\text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \\ \text{F1-Score} &= 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \end{aligned} \tag{15}$$

Precision, recall and F1 metrics are used in [5], [8] to estimate probabilistic object detector performance. One major issue of the above-mentioned evaluation metrics is that they do not take into account the values of the predicted uncertainty when determining if a detection is correct. As an example, a bounding box is considered correct if it achieves an IOU $\geq \delta_{iou}$ with a groundtruth box, a rule that does not depend on the bounding box's predicted uncertainty in any way, rendering the uncertainty prediction irrelevant to the evaluation.

2) Mean Average Precision: Average Precision (AP) was proposed by Everingham *et al*. [40] to evaluate the performance of object detectors. It is defined as the area under the continuous precision-recall (PR) curve, approximated through numeric integration over a finite number of sample points [40]. Mean AP (mAP) is the average of all AP values computed for every object category found in the test dataset. Mean AP can be averaged across multiple IOU thresholds $\delta_{iou} \in [0.5, \ldots, 0.95]$ as proposed in [122], a measure referred to as COCO mAP. Though mAP is the major evaluation metric in object detection, it too does not take the predictive

uncertainties into account. In fact, if two probabilistic object detectors predict bounding boxes with the same mean values but wildly different covariance matrices, they will have the same mAP performance. Nevertheless, the majority of recent work on probabilistic object detection [8], [11], [13]–[16], [114], [116]–[118] still use mAP as the only metric to provide a quantitative assessment of their proposed methods, emphasizing a secondary effect of accuracy improvement when integrating probabilistic detection methods, instead of focusing on the correctness of the output distribution. This leads to the need of better and more consistent metrics, which we highlight in Sec. V.

3) Probability Detection Quality: Hall *et al*. [132] proposed the Probability-based Detection Quality (PDQ) as a metric to measure the quality of 2D probabilistic object detection on images. PDQ was later used by [6], [12] for uncertainty evaluation. PDQ is designed to jointly evaluate semantic uncertainties and spatial uncertainties in image-based object detection. The semantic uncertainties are evaluated by matching the predicted classification scores with the ground truth labels for each pixel in images. The spatial uncertainties are encoded by covariance matrices, by assuming a Gaussian distribution on the top-right or the bottom-left corner of a bounding box. The optimal PDQ is achieved, when a predicted probability correlates the prediction error, for example, when a large spatial uncertainty correlates with an inaccurate bounding box prediction.

PDQ assigns every ground truth an optimal corresponding detection using the Hungarian algorithm, removing the dependency on IOU thresholding that is required for mAP and MUE. Furthermore, PDQ measures the probability mass assigned by the detector to *true positive detection* results, and is evaluated at a single classification score threshold requiring practitioners to filter low scoring output detection results prior to evaluation. PDQ also assumes 2D Gaussian corner distributions and cannot evaluate methods assuming a Laplace distribution such as [14], [15]. Finally, because of how the spatial quality is defined, PDQ can only be computed for 2D probabilistic detection results defined in image space, with no straightforward extensions available for 3D probabilistic object detectors.

4) Minimum Uncertainty Error: The Uncertainty Error (UE) was first proposed by Miller *et al*. [6] to evaluate probabilistic object detectors. UE can be thought of as the probability that a simple threshold-based classifier makes a mistake when classifying output detections into true positives and false positives using their predicted uncertainty estimates. UE ranges between 0 and 0.5, and as the uncertainty error approaches 0.5, using the predicted uncertainty estimates to separate true positives from false positives is no better than classifying with an unbiased coin flip. The best uncertainty error achievable by a detector over all possible thresholds is called the Minimum Uncertainty Error (MUE), and is used to compare probabilistic object detectors in [6], [12].

Similar to mAP, MUE requires an IOU threshold δ_{iou} to determine which detections are counted as true positives, and is therefore threshold-dependant. Furthermore, MUE is not affected by the scale of uncertainty estimates. In other words, rescaling or shifting the uncertainty values for all detections with the same value results in a constant MUE score. Therefore, MUE is only capable of providing information on how well the estimated uncertainty can be used to separate true positives from false positives, but not on the quality of estimated uncertainty.

5) Jaccard IOU: Recently, Wang *et al*. [116] propose Jaccard IOU (JIOU) as a probabilistic generalization of IOU. Unlike IOU which simply compares the (deterministic) geometric overlap between two bounding boxes, JIOU measures the similarity of their spatial distributions. Such distributions can be predicted by probabilistic object detection networks, or by inferring the uncertainties inherent in ground truth labels (which serve as the "references of probability distribution") [116]. In fact, JIOU simplifies to IOU when bounding boxes are assumed to follow simple uniform distributions. Similar to IOU, JIOU ranges within $[0, 1]$. It is maximized only when two bounding boxes have the same locations, same extents, as well as same spatial distributions. In general, JIOU provides a natural extension of IOU to evaluate probabilistic object detection. It also considers ambiguity and uncertainty inherent in ground truth labels, which are ignored by other evaluation metrics such as mAP and PDQ. However, a separate model is needed to devise accurate ground truth uncertainty in the labelling process, which has only been proposed for LiDAR point cloud [116]. Therefore, the use of JIOU is limited to evaluating LiDAR-based probabilistic object detection to date.

D. Sensor Modalities

Different sensors have different sensing properties and observation noises, and thus can affect the behaviours of (aleatoric) uncertainty in probabilistic object detection. Most studies to date employ LiDARs [7]–[10], [14]–[16], [105], [115] and RGB cameras [5], [6], [12], [13], [45], [107]–[109], [113], [127] in probabilistic object detection. Only Dong *et al*. [117] propose to model uncertainty in radar sensors. The LiDAR-based networks often deal with 3D detection for single object class or a few object classes, such as "Car", "Cyclist" and "Pedestrian". In comparison, the camera-based approaches focus on the multi-class 2D detection with more diverse classes. For example, Harakeh *et al*. [12] evaluate 2D probabilistic object detection with ten common road scene object categories in the BDD dataset [17], as well as 80 objects in the COCO dataset [122].

E. Applications and Use Cases of Predictive Uncertainty

The behaviours of epistemic and aleatoric uncertainties in a network are studied in [7], [16], [107], [109]. These works verify that epistemic uncertainty is related to the detections different from the training samples (open-set detections), whereas aleatoric uncertainty reflects complex noises inherent in sensor observations such as distance and occlusion. In addition, Bertoni *et al*. [109] study the task ambiguity in depth estimation with monocular images, and Wang *et al*. [116] study the uncertainty inherent in ground truth labels for LiDAR-based object detection.

Epistemic uncertainty is applied to detect out-of-distribution (OOD) objects in [5]. They find that thresholding the classification uncertainty captured by MC-dropout reduces detection errors for OOD objects when compared to the vanilla softmax function. Epistemic uncertainty is also used in [9] to select unseen samples in an active learning framework, in order to train LIDAR 3D object detectors with minimal human labeling effort.

Aleatoric uncertainty is mainly used to improve detection accuracy. This can be achieved by directly modelling uncertainty in the training loss (e.g. the attenuated regression loss shown in Eq. 4) [8], [11], [13], [115], such that networks learn to handle noisy data and enhance the robustness. It can also be achieved by incorporating uncertainty in the post-processing step when merging duplicate detections. For example, Choi *et al*. [114] rank detections during NMS based on location uncertainty in addition to classification scores. Meyer *et al*. [14] adapt the IoU threshold when merging detections to their location uncertainty instead of a pre-defined value for all detections. Finally, thresholding aleatoric uncertainty has been shown to effectively remove False Positive samples in [113].

IV. Comparative Study

A major challenge for new researchers and practitioners entering the probabilistic object detection domain is the lack of a consistent benchmark among methods presented in literature. This problem is evident when looking at Tab. II, where one can see that it is uncommon for state-of-the-art methods to use the same dataset and evaluation metric combinations, leading to difficulty in determining which method works best for autonomous driving. In this section, we attempt to alleviate this problem by performing a comparative study of the performance of state-of-the-art probabilistic object detectors using the same base network, datasets, and and multiple evaluation metrics.

In the sequel, Sec. IV-A summarizes the configurations of probabilistic object detectors used for the comparative study. Sec. IV-B and Sec. IV-C provide the experimental setup and implementation details, respectively. Sec. IV-D presents the main experimental results and our conclusions. Finally, Sec. IV-E compares the behaviours of aleatoric uncertainties in an image and a LiDAR-based object detectors.

A. Our Probabilistic Object Detectors

For a fair evaluation, we re-implement the various uncertainty estimation mechanisms described in Sec. III-B using a 2D image-based object detector, RetinaNet [18], as the basic network. The new network models the bounding box parameters as multivariate Gaussians with a diagonal covariance matrices, and uses the softmax function to predict the parameters of categorical distributions for object classification. The probabilistic extensions to RetinaNet are summarized in Tab. III.

To extend RetinaNet for modeling aleatoric uncertainty, we implement three different approaches: Loss Attenuation, BayesOD, and Output Redundancy, which are summarized in rows $1-3$ of Tab. III. **Loss Attenuation** uses the direct modeling approach for estimating the uncertainty in regression and classification outputs, by predicting the means and variances for bounding box regression variables and softmax logits. The network is trained using the attenuated regression losses as well as the modified classification loss to learn softmax logit variances, as presented in Sec. II-D3. Existing probabilistic object detectors which use this direct modelling approach are summarized in Sec. III-B2. **BayesOD** modifies the post-processing step of **Loss Attenuation** to replace non-maximum suppression (NMS) with data association followed by Bayesian Inference as proposed in [12]. Since only a single modification is required to formulate BayesOD from Loss Attenuation, comparing the performance of the two provides insights on how useful Bayesian inference is when compared to standard NMS. The final approach for estimating aleatoric uncertainty is **Output Redundancy**, which leaves the base network and detection head of the deterministic RetinaNet model intact, and only modifies the post-processing step to cluster redundant output detections (Sec. III-B2). To estimate category and bounding box uncertainty, Output Redundancy extracts sample statistics from clusters of output detections, avoiding the auxiliary variance predictions used in most probabilistic object detection approaches.

To model epistemic uncertainty, the uncertainty estimation method proposed in [5], [6] is implemented by incorporating dropout into the detection head of a deterministic RetinaNet model. We refer to this method as **Black Box** (row 8 of Tab. III), and follow the implementation described in [6] which clusters output after the NMS step. Since the **Black Box** and **Output Redundancy** implementations do not explicitly model the variance parameters of the bounding box output of RetinaNet, comparing the quality of their bounding box output uncertainty to methods that use direct modeling of the variance parameters should provide an indicator on the value of direct modeling in probabilistic object detection.

Finally, the probabilistic object detectors which jointly model epistemic and aleatoric uncertainties are summarized in rows $4-7$ of Tab. III. To model both types of uncertainty, we extend **Loss Attenuation** and **BayesOD** to perform stochastic dropout runs by modifying the detection head with dropout layers with two extensions: **Loss Attenuation + Dropout** and **BayesOD + Dropout**. **Loss Attenuation + Dropout** extends **Loss Attenuation** by performing multiple stochastic MC-dropout runs during inference, followed by the standard NMS. The same setting has been implemented in [7]. **BayesOD + Dropout** performs MC-dropout during inference, and then associates detections by a Gaussian Mixture Model to fuse aleatoric and epistemic uncertainty estimates according to Eq. 11 and Eq. 12. This method has been implemented previously in [12].

Additionally, we train independent **Loss Attenuation** models to construct deep ensembles [75]. We provide two implementations to fuse detections from ensemble results: **Pre-NMS Ensembles** fuse detections before the NMS step, and **Post-NMS Ensembles** performs data association and fusion after the NMS step. These ensemble approaches have not been attempted in literature, and can be considered a direct application of the uncertainty estimation mechanisms in [75]

TABLE III: A summary of uncertainty modeling methods used for the Base Network, Detection Head, and Post-processing stages of each of our implemented probabilistic object detectors.

Method	Unc. Type‡	Base Network	Detection Head	Post-processing
1. Output Redundancy	A	None	None	Data Association → Sample Statistics
2. Loss Attenuation	A	None	Reg and Cls variance	NMS
3. BayesOD	A	None	Reg and Cls variance	Data Association → Bayesian Fusion
4. Loss Attenuation + Dropout	A + E	None	Reg and Cls variance, MC-Dropout	NMS
5. BayesOD + Dropout	A + E	None	Reg and Cls variance, MC-Dropout	Data Association → Bayesian Fusion
6. Pre-NMS Ensembles	A + E	Ensembles	Reg and Cls variance, Ensembles	Data Association → Gaussian Mixture → NMS
7. Post-NMS Ensembles	A + E	Ensembles	Reg and Cls variance, Ensembles	NMS → Data Association → Gaussian Mixture
8. Black Box	E	None	MC-Dropout	NMS → Data Association → Gaussian Mixture

‡ A: Aleatoric uncertainty, E: Epistemic uncertainty

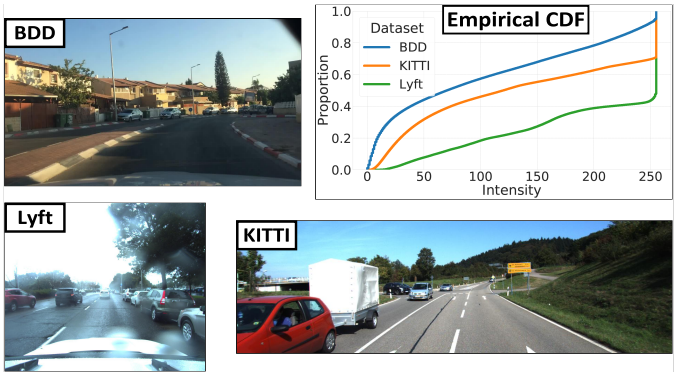


Fig. 6: Sample images from BDD, Lyft and KITTI datasets showing differences in aspect ratios, camera view points, and scene appearance between the three datasets. **Top Right:** Plot of the empirical cumulative distribution function (CDF) of the grayscale intensity values of all images in BDD, KITTI, and Lyft datasets. The CDF of KITTI is much closer to that of BDD than the CDF of Lyft.

to the object detection problem.

B. Experimental Setup

Recent work on evaluating uncertainty estimates of deep learning models for classification tasks [76] suggests the necessity of uncertainty evaluation under dataset shift. In this regard, we chose three common object detection datasets with RGB camera images for autonomous driving, in order to evaluate and compare our probabilistic object detectors:

- **Berkeley Deep Drive 100K (BDD) Dataset [17]** has $80,000$ image frames at a 1280×720 resolution, with $70,000/10,000$ training/validation data split. The BDD dataset was recorded in North America with diverse scene types such as city streets, residential areas, and highways, and in diverse weather conditions. The dataset contains an equal number of frames recorded during daytime and nighttime.
- **KITTI autonomous driving dataset [19]** contains $7,481$ image frames at a resolution of 1242×375. The dataset was recorded in Europe, only in clear weather and in daytime hours.
- **Lyft autonomous driving dataset [20]** has $158,757$ image frames at a 1224×1024 resolution. The Lyft Dataset was recorded in a single city (Palo Alto) in the US, and only contains 3D bounding box annotations. We generate 2D bounding box annotations for objects in the scene by projecting 3D bounding boxes using the Lyft Dataset API. Similar to KITTI, the Lyft dataset was captured during daytime and in clear weather.

All presented probabilistic object detectors are trained using the official training frames of BDD to detect seven common dynamic object categories relevant to autonomous driving: Car, Bus, Truck, Person, Rider, Bicycle, and Motorcycle. The detectors are then evaluated across all three datasets to quantify the uncertainty estimation quality with or without dataset shift.

Evaluation without Dataset Shift: First, we conduct comparative experiments without dataset shift, by testing the probabilistic object detectors on the official BDD validation dataset. This experimental setup corresponds to common object detection experiments in the literature, which quantify the performance of probabilistic object detectors on a test dataset with similar data distribution to the training set.

Evaluation with Dataset Shift: Both KITTI and Lyft have a different camera view point, image resolution, and image quality compared to BDD, which is visible when looking at example images from the three datasets in Fig 6. Furthermore, Fig 6 also shows that the empirical cumulative distribution function (CDF) of the greyscale intensity values of images from the KITTI dataset is much closer to the CDF of BDD than the CDF of Lyft, suggesting a smaller domain shift from BDD to KITTI than that from BDD to Lyft. The CDF plots in Fig 6 provide insight on the lack of exposure compensation in the Lyft dataset, where $\sim 60\%$ of pixels have an intensity greater than 250, and are therefore very bright (see Fig 6). The dataset shift from BDD to Lyft is further amplified as 2D bounding boxes are directly labeled on images for the BDD dataset, whereas 2D bounding boxes are generated by projecting 3D labels in the Lyft dataset. In summary, both KITTI and Lyft data is naturally shifted from the data in the BDD dataset, with KITTI having a lower magnitude of shift when compared to Lyft.

To quantify performance under dataset shift, probabilistic detection models trained **only on the BDD dataset** are tested

Method	mAP (%) ↑	PDQ (%) ↑	Cls NLL ↓	Reg NLL ↓	Frame Rate (FPS) ↑
Deterministic RetinaNet (Baseline)	28.62	-	0.70	-	22.01
Output Redundancy	26.90	18.40	0.91	2421.67	**15.21**
Loss Attenuation	28.54	33.29	0.70	15.27	14.97
BayesOD	28.65	**36.55**	0.70	13.06	9.75
Loss Attenuation + Dropout	27.51	32.34	0.74	15.36	2.61
BayesOD + Dropout	27.60	36.02	0.75	**13.03**	2.21
Pre-NMS Ensembles	**29.23**	33.21	0.71	15.24	6.06
Post-NMS Ensembles	29.18	32.47	**0.69**	15.23	3.35
Black Box	28.10	18.22	**0.69**	2806.52	2.45

TABLE IV: An evaluation with no dataset shift by testing detectors on the BDD validation dataset. Results are averaged over all seven dynamic object categories. Note that for NLL, lower values mean better performance.

Method	BDD	KITTI	Lyft
Deterministic RetinaNet (Baseline)	40.63	37.11	5.33
Output Redundancy	38.75	34.99	5.21
Loss Attenuation	41.09	39.15	5.96
BayesOD	41.33	39.32	**6.01**
Loss Attenuation + Dropout	40.39	38.90	5.97
BayesOD + Dropout	40.52	38.99	5.98
Pre-NMS Ensembles	41.85	39.50	5.953
Post-NMS Ensembles	**41.95**	**39.52**	5.949
Black Box	40.53	37.01	5.231

TABLE V: Mean average precision (mAP) results of evaluation under dataset shift. The mAP scrors are computed using the car and pedestrian categories on the BDD, Lyft, and KITTI datasets.

on 7,481 frames from the **KITTI** [19] dataset and 5,000 randomly chosen frames from the **Lyft** [20] autonomous driving dataset. The results on Lyft and KITTI are then compared against the results on the BDD validation dataset. Here, we only evaluate on the Car and Person object classes, as they are the only two classes that have the same definition in all three datasets.

C. Training and Inference

For all probabilistic object detectors mentioned above, we follow the training setup proposed in the original RetinaNet paper [18], using the authors' original open source implementation provided by the Detectron2 [133] 2D object detection library. More specifically, we use RetinaNet with a Resnet-50 backbone and a feature pyramid network (FPN). We train all models on the BDD training dataset split using a batch size of 4 for $90k$ iterations using stochastic gradient descent with 0.9 momentum. We use a base learning rate of 0.0025 which is dropped by a factor of 10 at 60,000 and then at 80,000 iterations. For methods requiring dropout, the dropout rate of 0.1 is selected. We find out that using higher dropout rates resulted in up to 10% drop in mAP when compared to the original RetinaNet model.

During inference, we use an NMS threshold of 0.5 for methods requiring NMS. For methods requiring data association, we use the Basic Sequential Algorithmic Scheme (BSAS) with a spatial affinity threshold of an IOU of 0.9 as recommended in [6]. For methods requiring dropout, we perform 10 stochastic dropout runs, as suggested by [57]. Since the deep ensemble method is computationally very expensive, we use an ensemble of 5 independently trained but identical models, the minimum number recommended in [75].

D. Experimental Results and Discussions

This section shows the key experimental results and findings from our proposed probabilistic object detectors. We first summarize the evaluation results in scenarios with and without dataset shift, and then list our findings with in-depth discussions.

General Results and Evaluation Metrics

Tab. IV presents the results of our comparative experiments on BDD validation data split without dataset shift, where metrics are averaged over the seven dynamic object categories of interest. We quantify the performance of probabilistic object detection using the mean Average Precision (mAP) and Probability Detection Quality (PDQ) metrics introduced in Sec. III-C, as well as the negative log likelihood estimates in Sec. II-E4. The negative log likelihood (NLL) is evaluated for both the estimated category and bounding box distributions of true positive detections, which have IOU > 0.7 scores with ground truths in the scene. This IOU threshold of 0.7 was chosen following the standard threshold set by the KITTI and BDD datasets when evaluating 2D object detection. Tab. IV also shows the runtime of each probabilistic object detection method on a machine with an Nvidia Titan V GPU and Intel i3770k CPU, in order to show the potential for deploying those methods on autonomous vehicles.

The first row of Tab. IV shows that our deterministic RetinaNet baseline achieves 28.62% mAP, an acceptable value that conforms to the 28% – 30% mAP range reported by baselines provided in the original BDD dataset paper [17]. We notice that all implemented probabilistic extensions to RetinaNet are less than 2% mAP difference compared to the

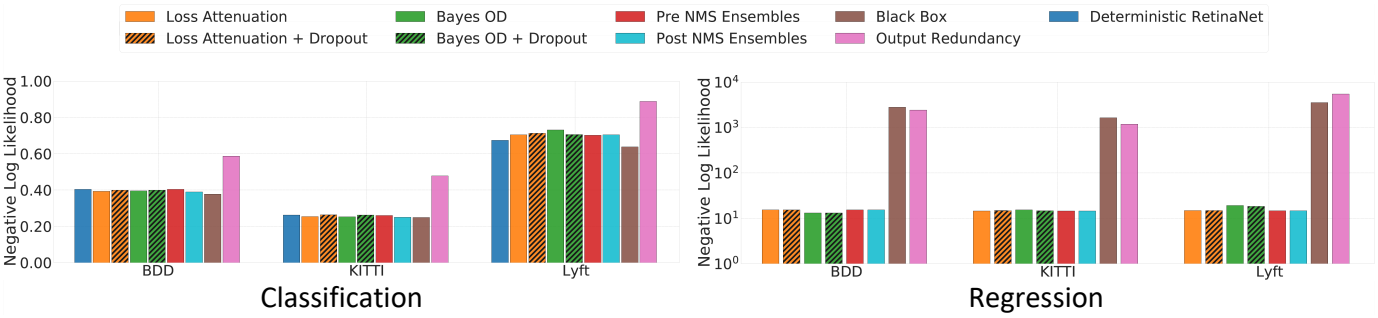


Fig. 7: Bar plots of the **negative log likelihood** of the category classification and bounding box regression probabilistic output for various probabilistic detection methods implemented with the RetinaNet architecture as a backend.

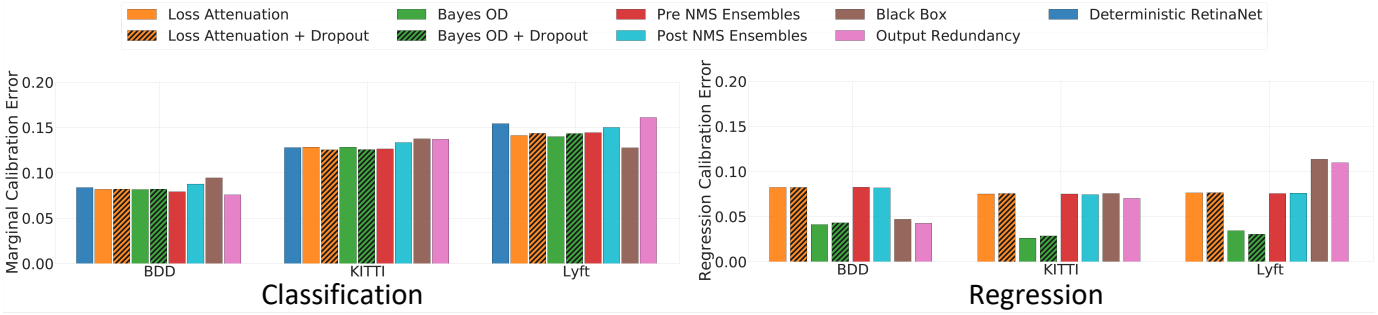


Fig. 8: Bar plots of the **calibration errors** of the category classification and bounding box regression probabilistic output for various probabilistic detection methods implemented with the RetinaNet architecture as a backend.

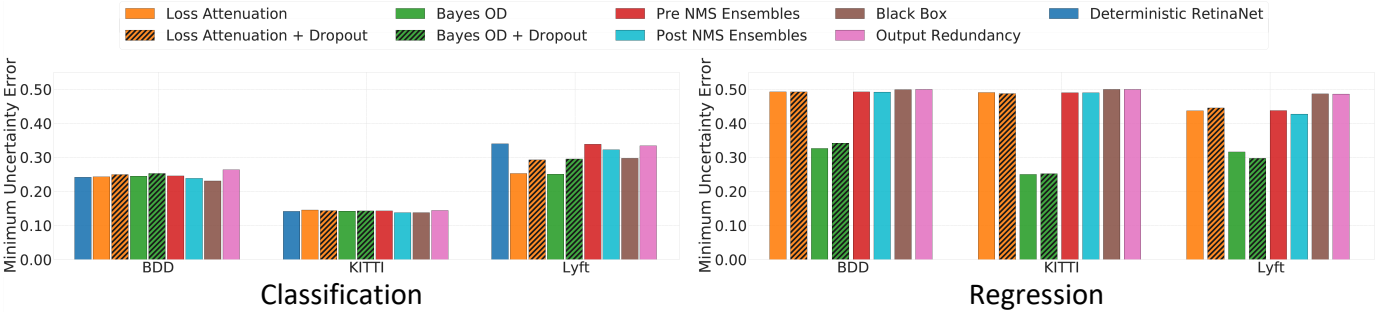


Fig. 9: Bar plots of the **minimum uncertainty errors** for the category classification and bounding box regression probabilistic output for various probabilistic detection methods implemented with the RetinaNet architecture as a backend.

deterministic baseline. On the other hand, Tab. V presents the mAP of the implemented probabilistic object detectors under dataset shift, evaluated only on the Car and Person categories on the BDD validation, KITTI, and Lyft datasets. Tab. V shows that when we only consider the Car and Person categories for our evaluation, all detection methods show an increase of mAP values from $\sim 28\%$ to $\sim 40\%$ on the BDD dataset, because detecting two object categories is easier than seven categories. Compared to the mAP performance on the BDD dataset, a slight drop of $2-3\%$ mAP is observed when evaluating on the KITTI dataset, as well as a substantial drop of over 30% mAP on the Lyft dataset. This is because the domain shift from BDD to Lyft data is much stronger than from BDD to KITTI data, as discussed in Sec. IV-B.

Tab. IV shows mAP to be correlated to the classification negative log likelihood (NLL). As an example, the method that achieves the lowest mAP in Tab. IV, Output Redundancy, is seen to have the highest classification NLL and as such the worst classification performance. On the other hand, Tab. IV shows mAP to have no correlation to the regression NLL. As an example, Tab. IV shows that Black Box has a regression NLL that is two orders of magnitude larger than other probabilistic detectors, while still performing on par with other probabilistic object detectors on mAP. Unlike mAP, we notice a direct correlation between PDQ and regression NLL, where the lowest PDQ is achieved by methods with the highest regression NLL (Black Box and Output Redundancy), while the highest PDQ is achieved by the methods with the lowest regression NLL. However, the magnitude of increase in PDQ between probabilistic object detectors is not proportional to the magnitude of decrease in regression NLL. As an example, Tab. IV shows that a reduction of 384.85 in regression NLL when comparing Output Redundancy to Black Box translates into an increase of 0.18 in PDQ.

Fig. 7, Fig. 8, and Fig. 9 show bar plots of the regression and classification NLL, calibration errors, and minimum uncertainty errors [12] respectively. Those uncertainty estimation values are averaged over the Car and Person categories from the BDD validation, KITTI, and Lyft datasets. For evaluating calibration performance, we use the Marginal Calibration Error (MCE) metric for category classification and Expected Calibration Error (ECE) for bounding box regression (both metrics have been introduced in Sec. II-E3). Regarding on the uncertainty estimation in classification, Fig. 7 (left) shows that all probabilistic object detectors exhibit a lower classification NLL scores on KITTI, but higher classification NLL scores on Lyft when compared to those on BDD. Even though we train our probabilistic object detectors using BDD data, probabilistic object detectors are seen to have lower NLL scores on KITTI than on the BDD validation dataset, implying higher confidence in correct detections. However, Fig. 8 (left) shows that the classification marginal calibration error increases to similar values on both KITTI and Lyft when compared to the marginal calibration error on the BDD validation set. We conclude that although our probabilistic object detectors provide sharper categorical distributions on KITTI compared to BDD, the distributions of KITTI detections are worse calibrated than those of BDD detections. Fig. 9 (left) also shows that the classification minimum uncertainty error follows the same trend as the classification negative log likelihood in Fig. 7 (left) for all evaluated probabilistic object detectors. Regarding on the uncertainty estimation in regression, the right plots from Fig. 7, Fig. 8, and Fig. 9 show that regression metrics have little difference under dataset shift. The reason for this phenomenon might be that we only evaluate well-localized regression outputs, where regressed bounding boxes are required to achieve an IOU ≥ 0.7 with ground truth boxes.

Finally, looking at the runtime of each probabilistic object detector in the last column of Tab. IV, we notice that all probabilistic extensions necessarily introduce a drop in frame rate when compared to deterministic RetinaNet. However, the drop introduced by methods that estimate aleatoric uncertainty such as Loss Attenuation is much lower than the drop introduced by parameter-sampling methods that estimate epistemic uncertainty (MC-Dropout and Deep Ensembles).

Comparison of each Method and Discussions

In the following, we compare each probabilistic object detection methods in detail, and highlight the most prominent takeaways from our comparative studies.

1. Using mAP as the only evaluation metric is not sufficient for quantifying the quality of predicted uncertainty in probabilistic object detectors. It is important to emphasize this point due to the widespread practice of using only mAP as an evaluation metric when comparing state-of-the-art probabilistic object detection (See Tab. II). mAP is clearly not related to the quality of uncertainty estimates provided by probabilistic object detectors, a quality that can be quantitatively observed in Tab. V. In this table, the mAP scores among different probabilistic object detection methods are comparable, with only $2-3\%$ mAP differences. However, the uncertainty scores (such as regression NLL) vary significantly, indicating very different uncertainty estimation performance. Taking a closer look at the performance of BayesOD and Pre-NMS ensembles, the table shows that BayesOD outperforms Pre-NMS ensembles on regression NLL, classification NLL and PDQ, but under-performs the same method in mAP, resulting different rankings of BayesOD and Pre-NMS ensembles. Since mAP is important for evaluating the performance of deterministic object detectors, we argue that a good probabilistic object detector should maintain competitive mAP when compared to deterministic counterparts. Therefore, we suggest that researchers use NLL, calibration errors, and PDQ alongside mAP for comparing and ranking probabilistic object detectors.

2. Estimating regression uncertainty through direct modeling is essential for high quality predictive distributions. Tab. IV shows that Black Box and Output Redundancy are a have a $\sim 50\%$ drop in PDQ scores when compared to the rest of the implemented methods. Tab. IV also shows that the regression NLL values of Black Box and Output Redundancy are two orders of magnitude larger than all other implemented probabilistic detectors. This observation is consistent when looking at results under dataset shift in Fig. 7, where regression NLL values of Black Box and Output Redundancy are seen to be around $\sim 10^3$ on BDD, KITTI, and Lyft data while the remaining probabilistic detectors have a regression NLL values that is closer to 10^1. In addition, Fig. 8 shows that the regression calibration error of both Black Box and Output Redundancy is around 0.04, on par with the best calibrated method, BayesOD, on the BDD dataset. However, the regression calibration error is seen to increase as the magnitude of the dataset shift increases, where Black Box and Output Redundancy become the methods with the highest regression calibration error on the Lyft dataset.

Tab. III shows that the common design decision in both Black Box and Output Redundancy is the lack of direct modeling of uncertainty in their detection head. Instead, both Black Box and Output Redundancy methods use redundant output detections to compute estimates of the sample covariance for regression output. Theoretically, a 4×4 diagonal sample covariance in 2D image space would require 2 *independent* output boxes to be estimate, and much more if this estimation is to be reliable. In addition, sample covariance estimates are highly sensitive to outliers [134]. Spatial clustering approaches used by the Black Box and Output Redundancy by definition need to sacrifice localization accuracy to obtain larger cluster sizes, increasing the probability of having outliers as cluster members. The Black Box method uses a low number of redundant boxes that are clustered after the non-maximum suppression (NMS) stage for regression uncertainty estimation, which is seen in Table IV as a higher regression NLL than that of Output Redundancy. On the other hand, Output Redundancy uses a large number of redundant output boxes collected before NMS, resulting in a better regression NLL at the expense of a much worse classification NLL, because low scoring detections are included in averaging their classification scores. For these reasons, we recommend the direct modeling approach for explicitly estimating the uncertainty of regressed

bounding boxes when designing probabilistic object detectors.

3. Estimating classification variance using loss attenuation provides no benefits over baseline classification from deterministic RetinaNet. Unlike the benefits offered by directly modeling the regression uncertainty, we found little benefits from modeling the classification variance using the formulation proposed by Kendall *et al*. [57] (and introduced in Sec. II-D3). Tab. IV shows that methods that explicitly estimate the variance of class logit vectors output by the network prior to the softmax output layer, such as Loss Attenuation, have the same classification NLL scores as the vanilla network softmax classification output distribution when tested with no dataset shift.

Under dataset shift, modeling the variance parameters for the logit vector results in mixed results when looking at NLL in Fig. 7, where Loss Attenuation showing a lower NLL value on the KITTI dataset but higher NLL on the Lyft dataset when compared to deterministic RetinaNet. On the other hand, Fig. 8 and 9 show that Loss Attenuation does have a lower classification calibration error and a lower classification minimum uncertainty error (MUE) on all three datasets when compared to deterministic RetinaNet. It should be noted that the improvement in performance observed using Loss Attenuation under domain shift is not substantial enough to validate the modifications to deterministic RetinaNet required to enable direct modeling of classification uncertainty. As such we find that the probability vector output of deterministic RetinaNet trained using standard cross entropy is sufficient to provide predictive uncertainty for the classification component of object detection without explicit modeling of the variance of the logits vector.

4. Bayesian Inference provides better regression uncertainty qualities when compared to standard NMS. This phenomenon has been previously shown in [12], which employs Bayesian Inference to fuse information from cluster members instead of selecting the detection with the highest score in the standard NMS. Tab. III shows that BayesOD follows the same probabilistic detection architecture as Loss Attenuation, with the only difference that BayesOD replaces NMS with the Bayesian Fusion. Tab. IV shows that with this simple change, BayesOD gains 3% in PDQ and reduces the regression NLL by 2.24 points compared to Loss Attenuation. Similarly, Fig. 8 shows that BayesOD has the lowest regression calibration error among all methods. Fig. 9 shows that BayesOD is the only method with a regression minimum uncertainty error substantially lower than 0.48, implying that the regression entropy output from BayesOD can be used to distinguish false positive detections from true positive detections. Unfortunately, the improvement in regression uncertainty estimation performance comes at a cost to runtime. Table IV shows that replacing NMS with Bayesian Inference in BayesOD causes a FPS from 15.21 to 9.75 when compared to Loss Attenution, which corresponds to approx. 37 ms more inference time. We therefore recommend that Bayesian Inference as a replacement for standard NMS, if better calibrated bounding box distributions are desired and the cost of additional computational can be compromised.

5. Sampling-based epistemic uncertainty estimation does not show big impact on the quality of estimated predictive uncertainty. Tab. IV and Tab. V show Loss Attenuation and BayesOD to exhibit a mAP drop when MC-Dropout is added to model epistemic uncertainty. On the other hand, both Pre-NMS and Post-NMS variants of ensembles are seen to provide a slight improvement to mAP both for evaluation without dataset shift in Tab. IV and for evaluation with dataset shift in Tab. V. However, the difference of mAP is only within $2-3\%$ range.

	Distance		Occlusion	
Sensors	Reg. Unc.	Cls. Unc.	Reg. Unc.	Cls. Unc.
Image	0.090	0.223	**0.485**	0.147
LiDAR	**0.551**	**0.510**	0.126	**0.158**

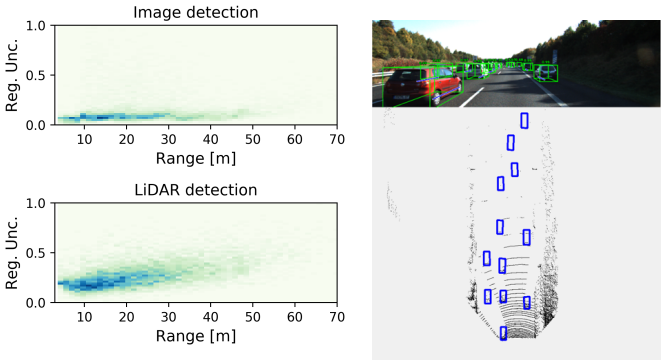
TABLE VI: The Pearson Correlation Coefficients (PCC) between aleatoric uncertainties and detection distances and occlusions, produced by a 2D image object detector and a 3D LiDAR object detector.

Fig. 7 shows that ensemble methods do not provide a substantial improvement in the quality of predicted uncertainty estimates compared to a single model, when evaluated using classification and the regression NLL. The same conclusion can be reached by observing calibration and MUE results in Fig. 8 and Fig. 9, respectively, where methods using ensembles exhibit similar performance to a single model trained with Loss Attenuation. On the other hand, Fig. 7, Fig. 8, and Fig. 9 show Loss Attenuation + Dropout and BayesOD + Dropout models to have a worse performance when compared to models trained with no dropout on NLL, calibration errors, and MUE respectively.

The reason behind the lack of significant improvement from Ensembles can be explained by looking at the size of the training dataset. As Kendall *et al.* suggested in [57], epistemic uncertainty can be explained away under large data situations such as our training on 70 K frames from the BDD dataset. MC-Dropout on the other hand has been shown to significantly under-perform ensembles and other methods on simple predictive uncertainty estimation tasks such as classification in [76], [135], so it does not come as a surprise that MC-Dropout enabled models do not provide improvement in the quality of predictive uncertainty. One final disadvantage of these sampling-based methods is how computationally expensive both MC-Dropout and ensemble-based methods are, with both methods having the slowest runtimes in Tab. IV. We therefore conclude that the current methods used for epistemic uncertainty estimation do not provide enough improvement in the quality of estimated uncertainty, while sacrificing large computational resources.

E. Aleatoric Uncertainty in Camera and LiDAR Perception

RGB camera images and LiDAR point clouds are two dominant sensing modalities in autonomous driving. They have unique sensing properties and observation noise characteristics, which fall in the category of aleatoric uncertainty. In this section, we compare aleatoric uncertainty in image perception and LiDAR perception. To do this, we conduct experiments



(a) (b)

Fig. 10: (a). The histograms of regression aleatoric uncertainties in image detections and LiDAR detections. (b). An illustration of object detections in RGB camera images and LiDAR point clouds in the bird's eye view.

on the "Car" objects in the KITTI dataset [19]. We collect 2D image detections from the RetinaNet (Loss Attenuation) introduced in Sec. IV-A, as well as their associated 3D LiDAR detections from ProbPIXOR [10] (cf. Tab. II). Tab. VI illustrates the correlations between aleatoric uncertainties in image or LiDAR perception and detection distance and occlusion, in terms of Pearson Correlation Coefficients (PCC). The aleatoric uncertaintes in regression and classification are measured by the Total Variance and Shannon Entropy metrics, respectively (cf. Sec. II-E). Aleatoric uncertainties (both classification and regression) in LiDAR detections are more correlated with detection distance than image detections with much larger PCC values. Fig. 10(a) supports this observation by plotting the distribution of regression uncertainties for all predictions (normalized to the same scale). Such uncertainty difference is due to the fact that the density of LiDAR point clouds are largely affected by distance, whereas the camera images do not directly provide the depth information (cf. Fig. 10(b)). Conversely, uncertainties in image detections (esp. in bounding box regression) are much more correlated with occlusion than those in LiDAR detections, because objects in images are often occluded and difficult to localize, while this is not the case in LiDAR point clouds (cf. Fig. 10(b)).

V. Challenges and Future Work

In this section, we discuss several challenges and directions for future work in probabilistic object detection.

A. Efficient Epistemic Uncertainty Estimation

Based on the papers reviewed, explicitly modelling epistemic uncertainty in probabilistic object detection usually requires a sampling mechanism such as MC-Dropout or Deep Ensembles, which are highly time-consuming and memory-intensive. For example, running MC-Dropout with 10 inferences has a speed of 2 – 3 FPS in Tab. IV, which is challenging for online autonomous driving. Recently, several works attempt to estimate epistemic uncertainties without sampling. For example, Postel *et al*. [56] propose to approximate epistemic uncertainties by an error propagation method. In general, error propagation methods (introduced in Sec. II-D4) have not yet been employed in probabilistic object detection. Furthermore, Sensoy *et al*. [23] and Amini *et al*. [78] have shown it is possible to directly predict the high-order conjugate priors of network output distributions (a Dirichlet prior for the multinomial distribution in classification [23], and a Inverse-Gamma prior for the Gaussian distribution in regression). They can therefore use single-shot inference to estimate epistemic uncertainties. Incorporating such error propagation and direct-modelling methods in probabilistic object detectors would be an interesting direction of future work.

B. Aleatoric Uncertainty Decomposition

Aleatoric uncertainty, defined as the observation noise inherent in sensor measurements, can be decomposed into several sources. For example, aleatoric uncertainty arises from varied weather conditions (e.g. consider night-time, fog, or rain scenarios in camera perception), sensor precision and quality, as well as data annotation errors or ambiguity (e.g. labeling a car with a mountain bike on its roof only as "Car" class, instead of as "Car" and "Bike" objects). As introduced in Sec. II-C, Recall that Czarnecki *et al*. [58] defined seven sources of uncertainty a perception module can model, as shown by Fig. 3. However, current probabilistic object detectors only model aleatoric uncertainty as a whole. It is an interesting future research direction to decompose aleatoric uncertainty and model each source of uncertainty separately. In this way, the uncertainty interpretability and detection performance may be potentially improved.

C. Uncertainty Propagation to Downstream Modules

The majority of works in probabilistic object detection focus on how to leverage uncertainty to improve the detection accuracy, especially in terms of Average Precision (AP), as we have discussed in Sec. III-C. However, object detection is only a building block in the long signal processing chain of the self-driving vehicle autonomy stack. Propagating the frame-by-frame detection uncertainties to the sequential downstream modules is expected to improve both the safety and robustness of the system as a whole. For example, multi-target tracking, an essential autonomous driving task, could benefit from estimated localization uncertainty of detected objects, which can be used to bootstrap object states in multi-target trackers based on Bayesian filtering [136]. Another task that could benefit from uncertainty estimates from object detection is the object-based simultaneous localization and mapping [137], where the predicted mean and covariance matrices from bounding boxes can be used as measurement updates. Finally, using uncertainty estimates from probabilistic object detection for behavior prediction of multiple agents in road scenes [138] or for driver assistance system [139] could be another interesting directions. We encourage further research on methods to incorporate predicted uncertainty from probabilistic object detection into the various tasks required by autonomous vehicles.

D. Better Evaluation of Probabilistic Object Detection

We have shown in Sec. IV that mean Average Precision (mAP) does not consider the quality of bounding box uncertainty in comparing probabilistic object detectors. Two objects with the same mean but different covariance matrices for their output bounding boxes will yield the same mAP. Given two predicted probability distributions, literature in probabilistic forecasting [140], [141] presents proper scoring rules as a way to evaluate which predicted distribution better matches the true data generating distribution. A proper scoring rule is only optimized, when a predicted distribution is equal to the ground truth distribution. Some examples of using proper scoring rules are the Brier Score (cf. Sec. II-E4) for classification tasks [76], and the Negative Log Likelihood (cf. Sec. II-E4) for both classification [76] and regression [75]. We recommend future works to evaluate probabilistic object detection using proper scoring rules, similar to our comparative study in Sec. IV. Furthermore, we argue that probabilistic object detection should be evaluated in context of the full autonomous vehicle stack (i.e. from perception and prediction to planning and action). It is a difficult task that requires access to an autonomous vehicle platform and a complete software stacks [142], [143] that support the integration of uncertainty estimates from probabilistic detectors in subsequent tasks. In this case, building autonomous vehicle simulators with high fidelity could help (e.g. [144], [145]). It is an important and challenging future research direction to build such a simulator that could take the predictive uncertainty from probabilistic object detectors as inputs, and evaluate probabilistic detection in a full software stack with novel metrics.

E. Ground truth Uncertainty for Datasets

Current open datasets in object detection (e.g. KITTI [19] and Waymo [131]) only provide ground truth boxes and object categorizes, which can be used to train deterministic object detectors. However, those datasets do not provide information to reflect the uncertainty (or ambiguity) of human annotations, and thus no ground truth information is available that can directly supervise the training of predictive uncertainty in probabilistic object detection. Ground truth uncertainty can be generated during the manual annotation process of object detection datasets, by quantifying the disagreement between different human annotators on the category and the bounding box location of the same object in the scene. Unfortunately, generating accurate ground truth uncertainty will require multiple iterations of labeling, a process that is expensive and time consuming. Recently, Wang *et al*. [116] make the first step to infer the label uncertainty with bounding boxes from LiDAR point clouds, with the help of a generative model. It would be an interesting direction to explore more alternative methods to generate ground truth uncertainty from a dataset, and benchmark probabilistic object detection.

VI. Conclusion

In this work, we present a survey and comparative study on probabilistic object detection using deep learning approaches in the field of autonomous driving. First, we provide an overview of uncertainty estimation in deep learning. Next, we systematically summarize existing methods in probabilistic object detection, and evaluate uncertainty estimation methods on a one-stage 2D image detector on three standard autonomous driving datasets. Finally, we discuss challenges and open questions in this research field.

The main conclusions from our comparative study are:

- The mean Average Precision (mAP) should not be used as the only evaluation metric for probabilistic object detectors, as it fails to capture the qualities of predictive uncertainty. To evaluate probabilistic object detection, it is recommended to use other evaluation metrics, such as Negative Log Likelihood (NLL) and calibration errors in conjunction with mAP.
- Estimating regression variance through the direct modeling approach is essential for good predictive uncertainty estimates.
- Estimating the variance in the softmax logit through the direct modeling approach does not show substantial improvement in detection accuracy or uncertainty quality.
- Bayesian inference [12] provides better regression uncertainty estimates, at a cost of slightly lower frame rate. It is recommended to replace Non-Maximum Suppression with Bayesian Inference, if additional computation cost can be afforded.
- Using epistemic uncertainty estimates from the sampling-based approaches (MC-Dropout and Deep Ensembles) only shows marginal improvement in uncertainty estimation, at the cost of high computation and memory.
- Different behaviours of aleatoric uncertainty are observed in the probabilistic object detectors using RGB camera images and LiDAR point clouds, indicating the necessity of modelling aleatoric uncertainty for different sensing modalities.

Open source code for benchmarking probabilistic object detection is made available at: https://github.com/asharakeh/pod_compare.git. We hope that our survey and the benchmarking code will become a useful tool for researchers and practitioners in developing robust perception in autonomous driving.

References

[1] L. Cosmides and J. Tooby, "Are humans good intuitive statisticians after all? rethinking some conclusions from the literature on judgment under uncertainty," *Cognition*, vol. 58, no. 1, pp. 1–73, 1996.

[2] J. Janai, F. Güney, A. Behl, A. Geiger *et al.*, "Computer vision for autonomous vehicles: Problems, datasets and state of the art," *Foundations and Trends in Computer Graphics and Vision*, vol. 12, no. 1–3, pp. 1–308, 2020.

[3] R. McAllister, Y. Gal, A. Kendall, M. Van Der Wilk, A. Shah, R. Cipolla, and A. V. Weller, "Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning," in *Int. Joint Conf. on Artificial Intelligence*, 2017.

[4] O. Willers, S. Sudholt, S. Raafatnia, and S. Abrecht, "Safety concerns and mitigation approaches regarding the use of deep learning in safety-critical perception tasks," in *International Conference on Computer Safety, Reliability, and Security*, 2020.

[5] D. Miller, L. Nicholson, F. Dayoub, and N. Sünderhauf, "Dropout sampling for robust object detection in open-set conditions," in *IEEE Int. Conf. Robotics and Automation*, 2018.

[6] D. Miller, F. Dayoub, M. Milford, and N. Sünderhauf, "Evaluating merging strategies for sampling-based uncertainty techniques in object detection," in *IEEE Int. Conf. Robotics and Automation*, 2019.

[7] D. Feng, L. Rosenbaum, and K. Dietmayer, "Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection," in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2018, pp. 3266–3273.

[8] D. Feng, L. Rosenbaum, F. Timm, and K. Dietmayer, "Leveraging heteroscedastic aleatoric uncertainties for robust real-time lidar 3d object detection," in *IEEE Intelligent Vehicles Symp.*, 2019.

[9] D. Feng, X. Wei, L. Rosenbaum, A. Maki, and K. Dietmayer, "Deep active learning for efficient training of a lidar 3d object detector," in *IEEE Intelligent Vehicles Symp.*, 2019.

[10] D. Feng, L. Rosenbaum, C. Gläser, F. Timm, and K. Dietmayer, "Can we trust you? on calibration of a probabilistic object detector for autonomous driving," *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2020.

[11] D. Feng, Y. Cao, L. Rosenbaum, F. Timm, and K. Dietmayer, "Leveraging uncertainties for deep multi-modal object detection in autonomous driving," *IEEE Intelligent Vehicles Symp.*, 2020.

[12] A. Harakeh, M. Smart, and S. L. Waslander, "Bayesod: A bayesian approach for uncertainty estimation in deep object detectors," in *IEEE Int. Conf. Robotics and Automation*, 2020.

[13] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding box regression with uncertainty for accurate object detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2019.

[14] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "LaserNet: An efficient probabilistic 3d object detector for autonomous driving," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2019.

[15] G. P. Meyer and N. Thakurdesai, "Learning an uncertainty-aware object detector for autonomous driving," *arXiv preprint arXiv:1910.11375*, 2019.

[16] S. Wirges, M. Reith-Braun, M. Lauer, and C. Stiller, "Capturing object detection uncertainty in multi-layer grid maps," in *IEEE Intelligent Vehicles Symp.*, 2019.

[17] F. Yu, H. Chen, X. Wang, W. Xian, Y. Chen, F. Liu, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving dataset for heterogeneous multitask learning," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.

[18] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018.

[19] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the KITTI vision benchmark suite," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2012.

[20] R. Kesten, M. Usman, J. Houston, T. Pandya, K. Nadhamuni, A. Ferreira, M. Yuan, B. Low, A. Jain, P. Ondruska *et al.*, "Lyft Level 5 AV Dataset 2019," *https://level5.lyft.com/dataset*, 2019.

[21] D. J. C. MacKay, "A practical Bayesian framework for backpropagation networks," *Neural Computation*, vol. 4, no. 3, pp. 448–472, 1992.

[22] R. M. Neal, "Bayesian learning for neural networks," Ph.D. dissertation, University of Toronto, 1995.

[23] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," in *Advances in Neural Information Processing Systems*, 2018.

[24] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *Int. Conf. Machine Learning*, 2015.

[25] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems*, 2011.

[26] S. Depeweg, J. Hernandez-Lobato, F. Doshi-Velez, and S. Udluft, "Decomposition of uncertainty in bayesian deep learning for efficient and risk-sensitive learning," in *Int. Conf. Machine Learning*, 2018.

[27] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.

[28] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Int. Conf. Machine Learning*, 2011.

[29] T. Chen, E. Fox, and C. Guestrin, "Stochastic gradient hamiltonian monte carlo," in *Int. Conf. Machine Learning*, 2014, pp. 1683–1691.

[30] X. Chen, J. D. Lee, X. T. Tong, Y. Zhang *et al.*, "Statistical inference for model parameters in stochastic gradient descent," *The Annals of Statistics*, vol. 48, no. 1, pp. 251–273, 2020.

[31] S. Mandt, M. D. Hoffman, and D. M. Blei, "Stochastic gradient descent as approximate bayesian inference," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 4873–4907, 2017.

[32] W. J. Maddox, P. Izmailov, T. Garipov, D. P. Vetrov, and A. G. Wilson, "A simple baseline for bayesian uncertainty in deep learning," in *Advances in Neural Information Processing Systems*, 2019.

[33] H. Ritter, A. Botev, and D. Barber, "A scalable laplace approximation for neural networks," in *6th International Conference on Learning Representations (ICLR)*, 2018.

[34] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, University of Cambridge, 2016.

[35] A. Kendall and R. Cipolla, "Modelling uncertainty in deep learning for camera relocalization," in *IEEE Int. Conf. Robotics and Automation*, 2016.

[36] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *Proceedings of the IEEE international conference on computer vision*, 2015.

[37] A. Kendall, V. Badrinarayanan, and R. Cipolla, "Bayesian SegNet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding," in *Proc. British Machine Vision Conf.*, 2017.

[38] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognition Letters*, vol. 30, no. 2, pp. 88–97, 2009.

[39] S. Song, S. P. Lichtenberg, and J. Xiao, "SUN RGB-D: A RGB-D scene understanding benchmark suite," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015.

[40] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The PASCAL visual object classes (VOC) challenge," *Int. J. Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[41] R. Mackowiak, P. Lenz, O. Ghori, F. Diego, O. Lange, and C. Rother, "Cereals-cost-effective region-based active learning for semantic segmentation," in *Proc. British Machine Vision Conf.*, 2018.

[42] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2016.

[43] E. Ilg, O. Ciçek, S. Galesso, A. Klein, O. Makansi, F. Hutter, and T. Brox, "Uncertainty estimates and multi-hypotheses networks for optical flow," in *Proc. Eur. Conf. Computer Vision*, 2018.

[44] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *Proc. Eur. Conf. Computer Vision*, 2012.

[45] S. Choi, K. Lee, S. Lim, and S. Oh, "Uncertainty-aware learning from demonstration using mixture density networks with sampling-free variance modeling," in *IEEE Int. Conf. Robotics and Automation*, 2018.

[46] J. Colyar and J. Halkias, "Us highway 101 dataset," *Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-07-030*, 2007.

[47] R. Michelmore, M. Wicker, L. Laurenti, L. Cardelli, Y. Gal, and M. Kwiatkowska, "Uncertainty quantification with statistical guarantees in end-to-end autonomous driving control," in *IEEE Int. Conf. Robotics and Automation*, 2020.

[48] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on Robot Learning*, 2017.

[49] A. Hu, F. Cotter, N. Mohan, C. Gurau, and A. Kendall, "Probabilistic future prediction for video scene understanding," *Proc. Eur. Conf. Computer Vision*, 2020.

[50] S. Walz, T. Gruber, W. Ritter, and K. Dietmayer, "Uncertainty depth estimation with gated images for 3d reconstruction," *arXiv preprint arXiv:2003.05122*, 2020.

[51] T. Gruber, F. Julca-Aguilar, M. Bijelic, and F. Heide, "Gated2depth: Real-time dense lidar from gated images," in *Proc. IEEE Conf. Computer Vision*, 2019.

[52] M. Danelljan, L. V. Gool, and R. Timofte, "Probabilistic regression for visual tracking," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.

[53] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2013, pp. 2411–2418.

[54] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *Proc. Eur. Conf. Computer Vision*, 2016.

[55] H. Kiani Galoogahi, A. Fagg, C. Huang, D. Ramanan, and S. Lucey, "Need for speed: A benchmark for higher frame rate object tracking," in *Proc. IEEE Conf. Computer Vision*, 2017.

[56] J. Postels, F. Ferroni, H. Coskun, N. Navab, and F. Tombari, "Sampling-free epistemic uncertainty estimation using approximated variance propagation," in *Proc. IEEE Conf. Computer Vision*, 2019.

[57] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Advances in Neural Information Processing Systems*, 2017.

[58] K. Czarnecki and R. Salay, "Towards a framework to manage perceptual uncertainty for safe automated driving," in *International Conference on Computer Safety, Reliability, and Security*. Springer, 2018, pp. 439–445.

[59] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "Salsanext: Fast semantic segmentation of lidar point clouds for autonomous driving," *arXiv preprint arXiv:2003.03653*, 2020.

[60] J. Gast and S. Roth, "Lightweight probabilistic deep networks," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2018.

[61] F. K. Gustafsson, M. Danelljan, and T. B. Schon, "Evaluating scalable bayesian deep learning methods for robust computer vision," in *CVPR Workshops*, 2020.

[62] A. Eldesokey, M. Felsberg, K. Holmquist, and M. Persson, "Uncertainty-aware cnns for depth completion: Uncertainty from beginning to end," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.

[63] N. Yang, L. v. Stumberg, R. Wang, and D. Cremers, "D3vo: Deep depth, deep pose and deep uncertainty for monocular visual odometry," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.

[64] G. Costante and M. Mancini, "Uncertainty estimation for data-driven visual odometry," *IEEE Transactions on Robotics*, 2020.

[65] W. H. Beluch, T. Genewein, A. Nürnberger, and J. M. Köhler, "The power of ensembles for active learning in image classification," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2018.

[66] E. Haussmann, M. Fenzi, K. Chitta, J. Ivanecky, H. Xu, D. Roy, A. Mittel, N. Koumchatzky, C. Farabet, and J. M. Alvarez, "Scalable active learning for object detection," in *IEEE Intelligent Vehicles Symp.*, 2020.

[67] K. Chitta, J. M. Alvarez, and A. Lesnikowski, "Large-scale visual active learning with deep probabilistic ensembles," *arXiv preprint arXiv:1811.03575*, 2018.

[68] G. P. Meyer, J. Charland, S. Pandey, A. Laddha, C. Vallespi-Gonzalez, and C. K. Wellington, "Laserflow: Efficient and probabilistic object detection and motion forecasting," *arXiv preprint arXiv:2003.05982*, 2020.

[69] M. Hudnell, T. Price, and J.-M. Frahm, "Robust aleatoric modeling for future vehicle localization," in *CVPR Workshops*, 2019.

[70] O. Makansi, E. Ilg, O. Cicek, and T. Brox, "Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2019.

[71] M. Segù, A. Loquercio, and D. Scaramuzza, "A general framework for uncertainty estimation in deep learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3153–3160, 2020.

[72] L. Tai, P. Yun, Y. Chen, C. Liu, H. Ye, and M. Liu, "Visual-based autonomous driving deployment from a stochastic and uncertainty-aware perspective," *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2019.

[73] Y. Gal, J. Hron, and A. Kendall, "Concrete dropout," in *Advances in Neural Information Processing Systems*, 2017.

[74] H. Blum, P.-E. Sarlin, J. Nieto, R. Siegwart, and C. Cadena, "Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving," in *CVPR Workshops*, 2019.

[75] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in Neural Information Processing Systems*, 2017.

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

[77] A. Malinin and M. Gales, "Predictive uncertainty estimation via prior networks," in *Advances in Neural Information Processing Systems*, 2018.

[78] A. Amini, W. Schwarting, A. Soleimany, and D. Rus, "Deep evidential regression," *arXiv preprint arXiv:1910.02600*, 2019.

[79] F. K. Gustafsson, M. Danelljan, G. Bhat, and T. B. Schön, "Energy-based models for deep probabilistic regression," in *Proc. Eur. Conf. Computer Vision*, August 2020.

[80] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Int. Conf. Machine Learning*, 2017.

[81] V. Kuleshov, N. Fenner, and S. Ermon, "Accurate uncertainties for deep learning using calibrated regression," *Thirty-fifth Interntional Conference on Machine Learning (ICML)*, 2018.

[82] A. Kumar, P. S. Liang, and T. Ma, "Verified uncertainty calibration," in *Advances in Neural Information Processing Systems*, 2019.

[83] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2009.

[84] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[85] A. Filos, S. Farquhar, A. N. Gomez, T. G. Rudner, Z. Kenton, L. Smith, M. Alizadeh, A. de Kroon, and Y. Gal, "A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks," in *Workshop on Bayesian Deep Learning (NeurIPS 2019)*, 2019.

[86] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2016.

[87] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Computer Vision*, 2016.

[88] A. Asvadi, L. Garrote, C. Premebida, P. Peixoto, and U. J. Nunes, "DepthCN: Vehicle detection using 3d-lidar and ConvNet," in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2017.

[89] M. Engelcke, D. Rao, D. Z. Wang, C. H. Tong, and I. Posner, "Vote3Deep: Fast object detection in 3d point clouds using efficient convolutional neural networks," in *IEEE Int. Conf. Robotics and Automation*, 2017.

[90] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems*, 2015.

[91] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proc. IEEE Conf. Computer Vision*, 2019.

[92] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," *Proc. Eur. Conf. Computer Vision*, 2020.

[93] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2017.

[94] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. Waslander, "Joint 3d proposal generation and object detection from view aggregation," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2018.

[95] M. Simon, S. Milz, K. Amende, and H.-M. Gross, "Complex-yolo: An euler-region-proposal for real-time 3d object detection on point clouds," in *Proc. Eur. Conf. Computer Vision*, 2018.

[96] B. Yang, W. Luo, and R. Urtasun, "PIXOR: Real-time 3d object detection from point clouds," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2018.

[97] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3d object detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2018.

[98] 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 *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.

[99] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[100] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2018.

[101] J. Fei, W. Chen, P. Heidenreich, S. Wirges, and C. Stiller, "Semanticvoxels: Sequential fusion for 3d pedestrian detection using lidar point cloud and semantic segmentation," in *IEEE Int. Conf. Multisensor Fusion and Integration*, 2019.

[102] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3d classification and segmentation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2017.

[103] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep sensor fusion for 3D bounding box estimation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2018.

[104] S. Shi, X. Wang, and H. Li, "Point-RCNN: 3d object proposal generation and detection from point cloud," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2019.

[105] D. Feng, C. Haase-Schuetz, L. Rosenbaum, H. Hertlein, F. Timm, C. Glaeser, W. Wiesbeck, and K. Dietmayer, "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges," *IEEE Trans. Intell. Transp. Syst.*, pp. 1–20, 2020.

[106] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama *et al.*, "Speed/accuracy trade-offs for modern convolutional object detectors," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, vol. 4, 2017.

[107] F. Kraus and K. Dietmayer, "Uncertainty estimation in one-stage object detection," in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2019.

[108] D. Miller, N. Sünderhauf, H. Zhang, D. Hall, and F. Dayoub, "Benchmarking sampling-based probabilistic object detectors." in *CVPR Workshops*, 2019.

[109] L. Bertoni, S. Kreiss, and A. Alahi, "Monoloco: Monocular 3d pedestrian localization and uncertainty estimation," in *Proc. IEEE Conf. Computer Vision*, 2019.

[110] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Third International Conference on Learning Representations (ICLR)*, 2015.

[111] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2016.

[112] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2015.

[113] M. T. Le, F. Diehl, T. Brunner, and A. Knol, "Uncertainty estimation for deep neural object detectors in safety-critical applications," in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2018.

[114] J. Choi, D. Chun, H. Kim, and H.-J. Lee, "Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving," in *Proc. IEEE Conf. Computer Vision*, 2019.

[115] H. Pan, Z. Wang, W. Zhan, and M. Tomizuka, "Towards better performance and more explainable uncertainty for 3d object detection of autonomous vehicles," in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2020.

[116] Z. Wang, D. Feng, Y. Zhou, W. Zhan, L. Rosenbaum, F. Timm, K. Dietmayer, and M. Tomizuka, "Inferring spatial uncertainty in object detection," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2020.

[117] X. Dong, P. Wang, P. Zhang, and L. Liu, "Probabilistic oriented object detection in automotive radar," in *CVPR Workshops*, 2020.

[118] Y. Lee, J.-w. Hwang, H.-I. Kim, K. Yun, and J. Park, "Localization uncertainty estimation for anchor-free object detection," *arXiv preprint arXiv:2006.15607*, 2020.

[119] Y. He, Z. Jianren Wang, Wang, and K. Jia, "Deep mixture density network for probabilistic object detection," 2020.

[120] D. Feng, L. Rosenbaum, F. Timm, and K. Dietmayer, "Labels are not perfect: Improving probabilistic object detection via label uncertainty," in *Eur. Conf. Computer Vision Workshops*, 2020.

[121] S. Wirges, T. Fischer, C. Stiller, and J. B. Frias, "Object detection and classification in occupancy grid maps using deep convolutional networks," in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2018.

[122] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Computer Vision*, 2014.

[123] J. McCormac, A. Handa, S. Leutenegger, and A. J. Davison, "Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation?" in *Proc. IEEE Conf. Computer Vision*, 2017.

[124] N. Sünderhauf, F. Dayoub, S. McMahon, B. Talbot, R. Schulz, P. Corke, G. Wyeth, B. Upcroft, and M. Milford, "Place categorization and semantic mapping on a mobile robot," in *IEEE Int. Conf. Robotics and Automation*. IEEE, 2016.

[125] 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," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.

[126] M. Braun, S. Krebs, F. Flohr, and D. M. Gavrila, "Eurocity persons: A novel benchmark for person detection in traffic scenes," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 8, pp. 1844–1861, 2019.

[127] Y. Chen, L. Tai, K. Sun, and M. Li, "Monopair: Monocular 3d object detection using pairwise spatial relationships," *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.

[128] N. Bodla, B. Singh, R. Chellappa, and L. S. Davis, "Soft-NMS-improving object detection with one line of code," in *Proc. IEEE Conf. Computer Vision*, 2017.

[129] S. Shao, Z. Zhao, B. Li, T. Xiao, G. Yu, X. Zhang, and J. Sun, "Crowdhuman: A benchmark for detecting human in a crowd," *arXiv preprint arXiv:1805.00123*, 2018.

[130] J. Wang, C. Xie, Z. Zhang, J. Zhu, L. Xie, and A. Yuille, "Detecting semantic parts on partially occluded objects," *Proc. British Machine Vision Conf.*, 2017.

[131] 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 *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020, pp. 2446–2454.

[132] 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 *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 1031–1040.

[133] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He, "Detectron," https://github.com/facebookresearch/detectron, 2018.

[134] P. J. Huber, *Robust statistics*. John Wiley & Sons, 2004, vol. 523.

[135] A. Ashukha, A. Lyzhov, D. Molchanov, and D. Vetrov, "Pitfalls of in-domain uncertainty estimation and ensembling in deep learning," *Eighth International Conference on Learning Representations (ICLR)*, 2020.

[136] C. Fruhwirth-Reisinger, G. Krispel, H. Possegger, and H. Bischof, "Towards data-driven multi-target tracking for autonomous driving," in *Computer Vision Winter Workshop (CVWW)*, 2020.

[137] L. Nicholson, M. Milford, and N. Sünderhauf, "Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 1–8, 2018.

[138] Y. Hu, W. Zhan, and M. Tomizuka, "Probabilistic prediction of vehicle semantic intention and motion," in *IEEE Intelligent Vehicles Symp.* IEEE, 2018.

[139] S. Liu, K. Koch, B. Gahr, and F. Wortmann, "Brake maneuver prediction – an inference leveraging rnn focus on sensor confidence," in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2019, pp. 3249–3255.

[140] J. Bröcker and L. A. Smith, "Scoring probabilistic forecasts: The importance of being proper," *Weather and Forecasting*, vol. 22, no. 2, pp. 382–388, 2007.

[141] T. Gneiting and A. E. Raftery, "Strictly proper scoring rules, prediction, and estimation," *Journal of the American statistical Association*, vol. 102, no. 477, pp. 359–378, 2007.

[142] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii, and T. Azumi, "Autoware on board: Enabling autonomous vehicles with embedded systems," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 2018, pp. 287–296.

[143] X. Huang, P. Wang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, "The apolloscape open dataset for autonomous driving and its application," *IEEE transactions on pattern analysis and machine intelligence*, vol. 42, no. 10, pp. 2702–2719, 2019.

[144] S. Manivasagam, S. Wang, K. Wong, W. Zeng, M. Sazanovich, S. Tan, B. Yang, W.-C. Ma, and R. Urtasun, "Lidarsim: Realistic lidar simulation by leveraging the real world," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.

[145] Z. Yang, Y. Chai, D. Anguelov, Y. Zhou, P. Sun, D. Erhan, S. Rafferty, and H. Kretzschmar, "Surfelgan: Synthesizing realistic sensor data for autonomous driving," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.