

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

Di Feng^{†1}, Ali Harakeh^{†2}, Steven Waslander², Klaus Dietmayer¹

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 either built with different network architectures and uncertainty estimation methods, or 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 practical uncertainty estimation methods 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

[†] 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

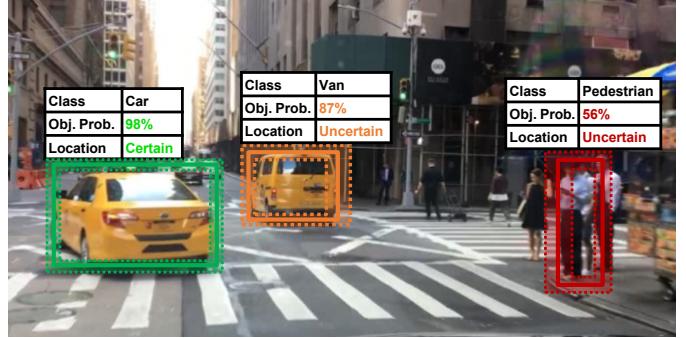


Fig. 1: A *conceptual* illustration of probabilistic object detection in an urban driving scenario. Each object is classified with a classification probability, and its bounding box is predicted with a confidence interval. The RGB camera image is from the BDD100k dataset [18].

(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].

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. When modeling probability in object detection, we need to estimate the probability an object belongs to the classes of interest (also called semantic uncertainty as introduced in [5]), as well as the probability distribution and the confidence interval of bounding box (i.e. spatial uncertainty [5]), as shown conceptually in Fig 1. In recent years, deep learning has become the de-facto approach in object detection, and many methods of modeling uncertainties in deep neural networks have been proposed [6]–[17]. 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 modeling 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 deep object detectors, and conduct a comparative study 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 [19], we benchmark several practical uncertainty estimation methods on three public autonomous driving datasets, namely, the KITTI object detection benchmark [20], the BDD100k Diverse Driving Video Database [18], and the Lyft Perception Dataset [21]. 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. BACKGROUND FOR 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 Modeling, and Error Propagation. Finally, Sec. II-E summarizes common evaluation metrics used in uncertainty estimation. Interested readers can also find a short summary of existing uncertainty estimation benchmarks in Appx. D.

A. Notation and Problem Formulation

Denote a labeled training dataset of N data pairs as $\mathcal{D} = \{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^N$, where \mathbf{x}_n is an input data sample in the domain \mathcal{X} , and \mathbf{y}_n its corresponding target value in the domain \mathcal{Y} . In classification, a target label can be one of the C classes, $y \in \{1, \dots, C\}$. In regression, a target value is usually a continuous vector with D dimensions denoted by $\mathbf{y} \in \mathbb{R}^D$. 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 $\hat{\mathbf{y}}$. Usually, deep neural networks are interpreted as point estimators with deterministic network

weights \mathbf{W} , and provide deterministic output predictions denoted by $\hat{\mathbf{y}} = f(\mathbf{x}, \mathbf{W})$. 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) [22], [23] 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:

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

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 by Direct Modeling (cf. 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) [24]. Multiple techniques for generating approximate solutions of the posterior distribution have been proposed [25]–[34]. Still, it is reported that approximation techniques in BNNs are highly sensitive to hyper-parameters, and hard to scale to large datasets and network architectures [33].

C. Epistemic and Aleatoric uncertainty

Based on the BNN framework, predictive uncertainty in deep neural networks can be decomposed into *epistemic* uncertainty and *aleatoric* uncertainty [35]. *Epistemic*, or model uncertainty, indicates how 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 not present in 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 a 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 in fitting training data, 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 uncertainties have been estimated in different components of autonomous driving such as semantic segmentation [36]–[38], optical flow [39], [40], depth estimation [35], [41]–[43], visual odometry [44]–[46], image annotation [47]–[50], visual tracking [51], trajectory prediction [52]–[55] and end-to-end perception [56]–[58]. Tab. VI in the appendix lists several uncertainty estimation applications in autonomous driving. Beyond epistemic and aleatoric uncertainty, Czarnecki *et al.* [59] define seven sources of perceptual uncertainty to be considered for the development and operation of a perception system, which is explained in Appx. A in detail.

D. Practical Methods for Uncertainty Estimation

In the following, we summarize four practical methods for predictive uncertainty estimation in deep learning: Monte Carlo Dropout (MC-Dropout), Deep Ensembles, Direct Modeling, 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

The MC-Dropout method proposed by Gal *et al.* [60] links dropout-based neural network training to Variational Inference (VI) in Bayesian Neural Networks (BNNs). They show that training a network with Stochastic Gradient Descent (SGD) with dropout is equivalent to optimizing a posterior distribution that approximates Eq. 1. The method is later extended in [61] to find the optimal dropout rate by treating the dropout probability as a hyperparameter. During test time, samples from the approximate posterior distribution are generated by performing inference 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:

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

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 Networks (BNNs), and is scalable to large datasets and network architectures such as in image classification [47] or semantic segmentation [42], [62]. However, MC-Dropout requires multiple stochastic runs during test time, usually 10 to 50 runs as shown in [60], making it still infeasible for real-time critical systems due to high computational cost.

2) Deep Ensembles

The method proposed by Lakshminarayanan *et al.* [63] estimates predictive probability using an ensemble of networks, where the outputs from each network are treated as independent samples from a mixture model. Each network in the ensemble uses 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, $\{\mathbf{W}_m\}_{m=1}^M$ their weights, and $p(\mathbf{y}|\mathbf{x}, \mathbf{W}_m)$ a prediction from the m -th network. The predictive probability is approximated by a uniformly-weighted mixture model with M components, given by:

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

The equation is similar to Eq. 2, as MC-Dropout can also be interpreted as an ensemble of networks [63]. In practice, an ensemble of 5 networks has been shown to be sufficient to approximate predictive probabilities [63], [64]. 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 [42].

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 $p(\mathbf{y} = c|\mathbf{x}, \mathbf{W}) = \hat{s}_c$, which is equivalent to a multinomial mass function. As for the regression probability, Gaussian distributions [35], [40] or Gaussian Mixture Models (GMM) [15], [65] are often used. For instance, it is often assumed that the target value \mathbf{y} in an one-dimensional regression problem is Gaussian distributed, given by: $p(\mathbf{y}|\mathbf{x}, \mathbf{W}) = \mathcal{N}(\mathbf{y}|\hat{\mu}(\mathbf{x}, \mathbf{W}), \hat{\sigma}^2(\mathbf{x}, \mathbf{W}))$. 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.

Training the network is achieved by maximum likelihood estimates, where \mathbf{W} are optimized to minimize the negative log likelihood $L(\mathbf{x}, \mathbf{W}) = -\log(p(\mathbf{y}|\mathbf{x}, \mathbf{W}))$. 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:

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

This function can be viewed as the standard L_2 loss 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, Kendall *et al.* [35] combine the softmax function with a Gaussian distribution to estimate classification uncertainty, 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 (More detailed explanation can be found in Appx. B). Furthermore, several works propose to estimate higher-order conjugate priors in addition to directly predicting the output probability distributions [24], [66], [67]. Finally, Gustafsson *et al.* [68] use a deep neural network to directly predict the conditional target density of an energy-based model.

Since Direct Modeling estimates uncertainty via single forward pass, it is much more efficient than MC-Dropout or Deep Ensembles. However, the network's output layers and the loss function need to be modified for uncertainty estimation. It has also been shown to produce miscalibrated probabilities in classification [60], [69] and regression [70].

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.* [37] 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 [60] using a single inference, significantly reducing the computation time. Similarly, Gast *et al.* [40] 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. Uncertainty Evaluation

When evaluating the quality of uncertainty, one needs to study the probability distributions that capture this uncertainty. Evaluating predicted probability distributions has been extensively studied in the context of pure inference problems [71], [72], and more recently in the context of machine learning applications [73], [74]. Geinting *et al.* [75] contend that the goal of a predicted probability distribution is to maximize the **sharpness** around the correct ground truth target, subject to **calibration**. Calibration [70], [76] is a joint property of the estimated predicted distribution as well as the correct ground truth events or values that materialized, and reflects their statistical consistency [75]. In simple terms, if a well-calibrated distribution assigns a 0.8 probability to an event (e.g. recognizing cars in object detection), the event should occur around 80% of the time. Sharpness on the other hand quantifies the concentration of the predictive distribution around the true materialized ground truth target and is a **property of the predictive distribution only**. As an example, if the correct ground truth class is “car”, the predicted probability distribution should ideally assign a probability of 1 to the car category. In the context of machine learning, good predicted probability distributions need to be sharp, but ideal predicted probability distributions should be both sharp and well-calibrated [75].

1) Measuring Calibration

A calibration plot is a common tool to evaluate the calibration of distributions generated from a probabilistic model. It draws the predicted probability from a model on the horizontal axis and its corresponding empirical probability on an evaluation dataset on the vertical axis, as shown in Fig. 2. A well-calibrated deep network produces a diagonal line. However, Guo *et al.* [69] and Kuleshov *et al.* [70] have empirically identified the over- or under-confident uncertainty in many classification and regression problems, and propose uncertainty recalibration techniques improve uncertainty estimation.

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

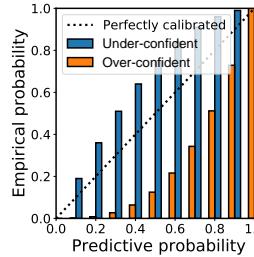


Fig. 2: A calibration plot. The horizontal axis represents the predictive probability, and the vertical axis the empirical probability. A miscalibrated network can make over-confident or under-confident predictions, as indicated in blue and orange curves, respectively.

network, denoted by p , are partitioned into T intervals (or quantiles), i.e. $0 < p_1 < \dots < p_t < \dots < 1$. They correspond to the values in the horizontal axis of Fig. 2. 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. 2. Drawing the calibration plots for regression probability distributions is more complicated, requiring the evaluation of cumulative distribution functions. We refer the reader to the work of Kuleshov [70] for a detailed description of how to draw calibration plots for regression predictive distributions. From the calibration plot we can derive several evaluation metrics. For example, Expected Calibration Error (ECE) [69] 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: $ECE = \sum_{t=1}^T \frac{N_t}{N_{eval}} |p_t - \hat{p}_t|$, 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 values indicating better uncertainty calibration performance. A regression alternative to ECE that can be extracted from regression calibration plots has been presented by Kuleshov in [70]. 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.* [76] 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.

2) Proper Scoring Rules

Unfortunately, measuring calibration is not enough to determine the quality of output probability distributions; we are required to also evaluate its sharpness, which we have described as the concentration of probability mass assigned by output probability distributions around the correct ground truth targets. We can evaluate both calibration and sharpness of probability distributions by using **proper scoring rules**. A scoring rule $S(p(\mathbf{y}|\mathbf{x}, \mathcal{D}), \mathbf{y}_n)$ is a function that maps a predicted probability distribution and the correct ground truth target to single scalar value. A lower value of a scoring rule usually signifies predicted probability distributions that are closer to the data generating distribution from which the ground truth target is sampled. A **proper scoring rule** is a scoring rule that is only minimized if the predicted probability distribution is exactly equal to the distribution that generated

the correct ground truth target. Note that we do not need access to the “theoretical” data generating distribution to compute a scoring rule, we only require a single sample from this distribution which is readily available in dataset \mathcal{D} . In addition, proper scoring rules have been shown in [77] to be sum of terms relating to calibration as well as sharpness, and as such proper scoring rules are capable of considering both when evaluating predictive distributions [78]. For more thorough arguments on the importance of using proper scoring rules for evaluating predicted probability distributions, we refer the reader to [74], [75], [77]–[79]. We now present two common proper scoring rules used in machine learning literature for evaluating predicted probability distributions [63], [64].

Negative Log Likelihood (NLL). NLL is a proper scoring rule to measure the quality of predicted probability distributions of a test dataset with N_{test} data points. NLL is calculated as: $NLL = -\frac{1}{N_{test}} \sum_{n=1}^{N_{test}} \log p(\mathbf{y}_n | \mathbf{x}_n, \mathcal{D})$, where \mathbf{x}_n is a test data point, and \mathbf{y}_n its corresponding ground truth label. NLL ranges in $(-\infty, +\infty)$, with a lower NLL score indicates a better fitting predictive distribution for that specific ground truth label. NLL has been extensively used in machine learning literature for evaluating output probability distributions for regression [63] and classification [64] tasks.

Brier Score (BS). A Brier Score [71] is another common proper scoring rule used to measure the quality of output probability distributions specifically in classification. The Brier score 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: $BS = \frac{1}{N_{test}} \sum_{n=1}^{N_{test}} \sum_{c=1}^C (\hat{s}_{n,c} - y_{n,c})^2$, 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 lower values indicating better uncertainty estimation.

We will use NLL, Brier Score, alongside calibration errors for evaluating both regression and classification probability distributions **of in-distribution detection results** predicted by our probabilistic detectors in Sec. IV-D. It is noteworthy to mention that there exist several non-proper scoring metrics to measure the magnitude of uncertainty, such as Mutual Information, Shannon Entropy, and Total Variance. Interested readers can find additional non-proper scoring metrics in Appx. C.

III. PROBABILISTIC OBJECT DETECTION

As defined in [5], probabilistic object detection aims to accurately detect objects, while estimating the **semantic (category classification)** and **spatial (bounding box regression)** uncertainties in each 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. Introduction

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 [60]). 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 predict probability distributions describing the category as well as the bounding box of objects in the scene [5]. Fig. 3 provides an overview of the general building blocks used 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, so as 2D and 3D object detection. Therefore, we do not distinguish between the two for the remainder of this paper.

A *base network* (Fig. 3, 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 [8], [80]–[83]. Typical base networks include VGG [84], ResNet [85], Inception [86], to name a few. A *detection head* (Fig. 3, Middle) takes the feature maps from the base network as inputs and predicts probabilistic detection outputs. Deep Ensembles [80], [82] and MC-Dropout [6], [7], [13], [17], [81]–[83] are usually used to model epistemic uncertainty in the detection head. On the other hand, the Direct Modeling 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 [8], [9], [11]–[17], [31], [80], [81], [83], [87]–[91] or the variance of output logits [12], [87]. It has to be noted that the variance of the logits does not need to be estimated for generating a categorical distribution describing object categories; any detector that employs softmax as an output layer while learning with cross-entropy loss captures aleatoric uncertainty in classification. Finally, a *post-processing stage* is used to suppress or merge redundant detection outputs as is common in deterministic object detectors. Tab. I summarizes the post-processing step in state-of-the-art probabilistic object detectors. The majority of methods employ 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 [15], [16], [88] modify NMS to take into account spatial uncertainty when deciding which redundant detection to remove,

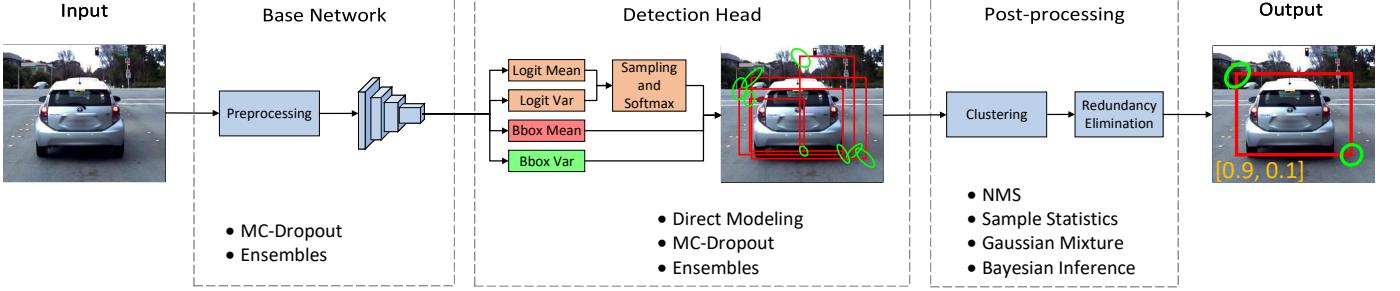


Fig. 3: Illustration of the key building blocks usually present in state-of-the-art probabilistic object detectors, including the base network, detection head and post-processing stages. A list of possible variants of each building block is also presented below the architectural diagram. The output detections 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).

or replace NMS with Bayesian Inference [13] to include information from all redundant detection outputs. Finally, [6], [7], [87] take the benefits of redundant detection outputs to compute sample statistics of probability distributions, instead of directly-modeling the output probability and discarding redundant detections.

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. I provides an overview of all methods we have reviewed.

1) Probabilistic Object Detectors with Epistemic Uncertainty

Several works model epistemic uncertainty with the MC-Dropout approach. Miller *et al.* [6] estimate the classification epistemic uncertainty by modifying SSD [92] deterministic backend. SSD's 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 affinity. For every cluster, the categorical probability distribution of a single object is modeled using Eq. (2). The same architecture is used in [7] for studying the effect of various clustering techniques on the quality of epistemic uncertainty in the object classification task. 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.* [82] avoid the clustering process when computing classification epistemic uncertainty. They build two detectors based on the Faster-RCNN [93] and SSD [92] architectures, each generates multiple samples at a certain anchor grid location by the MC-Dropout inference. For each detector separately, samples are used to compute the mean of the softmax classification output at every anchor location using Eq. 2 or Eq. 3. Uncertainty is then modeled using the entropy of the probability estimates from the softmax classification output. Finally, redundant detections are merged via the standard NMS operation. Experimental results show that deep ensembles outperform the MC-Dropout from a

single model on the two studied architectures, in terms of the predictive uncertainty estimation.

Finally, Feng *et al.* [10] use MC-Dropout [60] and Deep Ensemble [63] 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 Modeling approach has been used in various object detection architectures, including SSD [87], Faster-RCNN [9], [12], [14], FCOS [94], Point-RCNN [89], and PIXOR [11].

The softmax function is widely used to estimate classification probability, which corresponds to a multinomial mass function. Le *et al.* [87] and Feng *et al.* [12] assume a Gaussian distribution on the logit vectors, which is then transformed to a categorical distribution through the monte-carlo sampling process described in [35]. 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) [9], [13], [15], [88]. 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.* [89] transform the regression variables back to the eight-corner representation for 3D bounding boxes, and directly learn the corner uncertainty; Meyer *et al.* [15] place a mixture of Gaussians on each regression variable; Harakeh *et*

al. [13] learn a multivariate Gaussian distribution with the full covariance matrix for regression variables. The recent work by He *et al.* [95] propose a more generic probability distribution which considers both correlations and multi-modal behaviours. They use a multi-variate 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 (NLL), 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, [14], [16] introduce a probability distributions 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 groundtruth probability distribution, which results in more stable training and improved detection when compared to standard negative log likelihood loss [16], [96].

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.* [15] propose to group detections using mean-shift clustering over corners, and combine the bounding box uncertainty and classification scores in an uncertainty-aware NMS framework. Similarly, Choi *et al.* [88] 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 Modeling method, a unique approach based on output redundancy is proposed in [87] to model aleatoric uncertainty for both category classification and bounding box regression. 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. Two probability distribution describing each output detection can then be generated by computing the sample mean of the softmax output for the category, and the sample mean and variance of cluster members for the output bounding box. Though the output redundancy method has been shown in [13] to produce lower quality uncertainty estimates than the Direct Modeling method (measured with PDQ), it is time-efficient, and requires only 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 Modeling 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:

$$\hat{\mu}(\mathbf{x}) = \frac{1}{T} \sum_{t=1}^T \hat{\mu}(\mathbf{x}, \mathbf{W}_t) \quad (5)$$

$$\hat{\sigma}^2(\mathbf{x}) = \hat{\sigma}_e^2(\mathbf{x}) + \hat{\sigma}_a^2(\mathbf{x}) \quad (6)$$

$$\hat{\sigma}_e^2(\mathbf{x}) = \frac{1}{T} \sum_{t=1}^T (\hat{\mu}(\mathbf{x}, \mathbf{W}_t))^2 - (\hat{\mu}(\mathbf{x}))^2 \quad (7)$$

$$\hat{\sigma}_a^2(\mathbf{x}) = \frac{1}{T} \sum_{t=1}^T \hat{\sigma}^2(\mathbf{x}, \mathbf{W}_t) \quad (8)$$

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 $\hat{\mu}(\mathbf{x})$ and variance $\hat{\sigma}^2(\mathbf{x})$ 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. (6)-(8), we refer the reader to [63].

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.* [8] add dropout layers and perform multiple inferences only in the detection head of a Faster-RCNN object detector to estimate uncertainty for both category and bounding box of objects in the scene. Wirges *et al.* [17] 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.* [13] modify a 2D image detector based on RetinaNet [19], and incorporate MC-Dropout in the detection head. Kraus *et al.* [81] 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, [8], [81], [97] use standard NMS. Harakeh *et al.* [13] 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. I 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 positive and false positive counts [87] to complex metrics such as Probability-based Detection Quality (PDQ) [5]. The following section summarizes the common evaluation metrics, and discusses their properties when evaluating probabilistic object detection.

TABLE I: A summary of probabilistic object detectors.

Reference	Year	Sensor [†]	Dataset	Unc. Type [‡]	Modeling Method	Post-processing	Evaluation Metrics
Feng <i>et al.</i> [8]	2018	L	KITTI [20]	E+A	MC-Dropout, Direct Modeling	NMS	F-1 Score
Feng <i>et al.</i> [9] Le <i>et al.</i> [87]	2018	L C	KITTI [20]	A A	Direct Modeling Direct Modeling, Output Redundancy	NMS, Output Statistics	mAP True Positive/False Positive count
Miller <i>et al.</i> [6]	2018	C	COCO [98], SceneNet RGB-D [99], QUT Campus Dataset [100]	E	MC-Dropout	NMS, Clustering	Precision and Recall, Absolute Open-set Error, F1-Score
Choi <i>et al.</i> [88]	2019	C	BDD [18], KITTI [20]	A	Direct Modeling	Uncertainty-aware NMS	AP
Feng <i>et al.</i> [11]	2019	L	KITTI [20], NuScenes [101]	A	Direct Modeling	NMS	mAP, Expected Calibration Error
He <i>et al.</i> [14]	2019	C	COCO [98], Pascal-VOC [102]	A	Direct Modeling	NMS	mAP
Kraus <i>et al.</i> [81]	2019	C	EuroCity Persons [103]	E+A	MC-Dropout, Direct Modeling	NMS, Clustering	Log Average Miss Rate
Meyer <i>et al.</i> [15]	2019	L	KITTI [20], ATG4D [104]	A	Direct Modeling	Mean-shift clustering, Uncertainty-aware NMS	mAP
Meyer <i>et al.</i> [16]	2019	L	ATG4D [104]	A	Direct Modeling	Mean-shift clustering, Uncertainty-aware NMS	mAP, Calibration Plots, Label Uncertainty
Miller <i>et al.</i> [7]	2019	C	COCO [98], VOC [102], Underwater Scenes	E	MC-Dropout	NMS, Clustering	mAP, Uncertainty Error, AUPR, AUROC
Miller <i>et al.</i> [82]	2019	C	COCO [98]	E	MC-Dropout, Deep Ensembles	NMS, Clustering	PDQ
Wirges <i>et al.</i> [17]	2019	L	KITTI [20]	E+A	MC-Dropout, Direct Modeling	NMS, Clustering	mAP
Lee <i>et al.</i> [94]	2020	C	COCO [98]	A	Direct Modeling	Uncertainty-aware NMS	mAP
Chen <i>et al.</i> [105]	2020	C	KITTI [20]	A	Direct Modeling	Pairwise spatial constrain optimization	mAP
Dong <i>et al.</i> [91]	2020	R	Self-recorded data	A	Direct Modeling	Soft NMS [106]	mAP
Feng <i>et al.</i> [12]	2020	L+C	KITTI [20], NuScenes [101], Bosch	A	Direct Modeling	NMS	mAP
Harakeh <i>et al.</i> [13]	2020	C	COCO [98], VOC [102], BDD [18], KITTI [20]	E+A	MC-Dropout, Direct Modeling	Bayesian Fusion	mAP, Uncertainty Error, PDQ
He <i>et al.</i> [95]	2020	C	COCO [98], VOC [102], CrowdHuman [107], VehicleOcclusion [108]	A	Direct Modeling	Mixture of Gaussians, NMS	mAP, PDQ
Pan <i>et al.</i> [89]	2020	L	KITTI [20]	A	Direct Modeling	NMS	mAP
Wang <i>et al.</i> [90]	2020	L	KITTI [20], Waymo [109]	A	Direct Modeling	NMS	mAP, ROC curves, Jaccard-IOU

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

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 truth objects. Detections above a certain pre-defined IOU threshold, δ_{iou} , are considered as True Positives if they share the same class labels as the ground truth objects. Detections that are misclassified, or have an IOU below the δ_{iou} , are considered False Positives. False Negatives (FN) are further defined as the ground truth bounding boxes, which are either missed or not associated with predictions due to low IOU scores. The counts of TP and FP were used in [87] to evaluate their proposed probabilistic object detector. TP and FP scores were also used to construct the Receiver Operating Characteristic (ROC) curves in [7], [90].

Based on TP, FP, and FN, one can derive Precision, Recall, and the F1-Score as: Precision = TP/(TP+FP), Recall = TP/(TP+FN), F1-Score = 2 · Precision · Recall/(Precision+Recall). Precision, recall and F1 metrics are used in [6], [9] 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 $\text{IOU} \geq \delta_{iou}$ 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.* [102] 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 [102]. Mean AP (mAP) is the average of all AP values computed for every object category found in the test dataset. It can also be averaged across multiple IOU thresholds $\delta_{iou} \in [0.5, \dots, 0.95]$ as proposed in [98], a measure referred to as COCO mAP. Though mAP is the standard evaluation metric in object detection, it 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 [9],

[12], [14]–[17], [88], [90], [91], [94] 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-based Detection Quality

Hall *et al.* [5] 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 [13], [82] 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 object instance 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. 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 assumes 2D Gaussian corner distributions and cannot evaluate methods assuming a Laplace distribution such as [15], [16]. 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.* [7] 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 spatial 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 a random classifier. 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 [7], [13].

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.* [90] 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”) [90]. 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 approximate ground truth spatial uncertainty in the labeling process, which has only been proposed for LiDAR point cloud [90], [110]. Therefore, the use of JIoU is limited to evaluating LiDAR-based probabilistic object detection to date.

6) Proper Scoring Rules

To our surprise, none of the previous work on probabilistic object detectors use proper scoring rules for evaluating the quality of their predicted probability distributions. This observation was also noted in the recent work of Harakeh and Waslander [111], where it was argued that proper scoring rules are essential for a theoretically sound evaluation of predicted distributions from probabilistic object detectors. It was also shown in [111] that mAP, MUE, and PDQ **are not proper scoring rules**. For our experiments in Sec. IV-D, we will use NLL as a proper scoring rule for evaluating the quality of predicted category and bounding box probability distributions.

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 [8]–[11], [15]–[17], [80], [89] and RGB cameras [6], [7], [13], [14], [65], [81]–[83], [87], [105] in probabilistic object detection. Only Dong *et al.* [91] 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.* [13] evaluate 2D probabilistic object detection with ten common road scene object categories in the BDD dataset [18], as well as 80 object categories in the COCO dataset [98].

E. Applications and Use Cases of Predictive Uncertainty

The behaviours of epistemic and aleatoric uncertainties in a network are studied in [8], [17], [81], [83]. 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.* [83] study the task ambiguity in depth estimation with monocular images, and Wang *et al.* [90] 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 [6]. They find that thresholding the classification uncertainty captured by MC-Dropout reduces detection errors for OOD objects when compared to the standard softmax confidence. Epistemic uncertainty is also used in [10] 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 modeling uncertainty in the training loss (e.g. the attenuated regression loss shown in Eq. 4) [9], [12], [14], [89], 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.* [88] rank detections during NMS based on location uncertainty in addition to classification scores. Meyer *et al.* [15] 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 [87].

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. I, 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 common modifications described in Sections II and III that are used to extend deterministic object detectors to predict probability distributions. By using the same base network, datasets, and multiple evaluation metrics, we aim to have a controlled setup to perform a fair comparison. Specifically, we focus on 2D image-based detection for seven object categories. All networks are trained on the large-scale BDD100K dataset [18], and evaluated on the BDD100K, KITTI [20], and Lyft [21] datasets. Note that due to the lack of large-scale open datasets with diverse multiclass labels, we do not conduct experiments for 3D object detection and for LiDAR point cloud.

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 [19], 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. II.

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. II. **Loss Attenuation** uses **Direct Modeling** 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 (details cf. Appx. B). Existing probabilistic object detectors which use this Direct Modeling 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 [13]. 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, methods proposed in [6], [7] are implemented by incorporating dropout into the detection head of a deterministic RetinaNet model. We follow the implementation described in [7] which clusters output after the NMS step, treating the object detector as a **Black Box** (row 8 of Tab. II). 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. II. To model both types of uncertainty, we extend **Loss Attenuation** and **BayesOD** to perform stochastic dropout runs by modifying the detection head with dropout

TABLE II: 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	Unmodified	Unmodified	Data Association → Sample Statistics
2. Loss Attenuation	A	Unmodified	Reg and Cls variance	NMS
3. BayesOD	A	Unmodified	Reg and Cls variance	Data Association → Bayesian Fusion
4. Loss Attenuation + Dropout	A + E	Unmodified	Reg and Cls variance, MC-Dropout	NMS
5. BayesOD + Dropout	A + E	Unmodified	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	Unmodified	MC-Dropout	NMS → Data Association → Gaussian Mixture

‡ A: Aleatoric uncertainty, E: Epistemic uncertainty

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 [8]. **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. 5 and Eq. 6. This method has been implemented previously in [13].

Additionally, we train independent **Loss Attenuation** models to construct deep ensembles [63]. 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 + loss-attenuation approaches have not been attempted in literature, and can be considered a direct application of the uncertainty estimation mechanisms in [63] to the object detection problem.

B. Experimental Setup

Recent work on evaluating uncertainty estimates of deep learning models for classification tasks [64] 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: the **Berkeley Deep Drive 100K (BDD) Dataset** [18], the **KITTI autonomous driving dataset** [20], and the **Lyft autonomous driving dataset** [21]. 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: To quantify performance under dataset shift, probabilistic detection models trained **only on the BDD dataset** are tested on 7,481 frames from the

KITTI [20] dataset and 5,000 randomly chosen frames from the **Lyft** [21] 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. More detailed information on three datasets as well as the dataset shift from BDD to KITT and BDD to Lyft can be found in Appx. F.

C. Training and Inference

For all probabilistic object detectors mentioned above, we follow the training setup proposed in the original RetinaNet paper [19], using the authors’ original open source implementation provided by the Detectron2 [112] 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 [7]. For methods requiring dropout, we perform 10 stochastic dropout runs, as suggested by [35]. 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 [63].

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. III 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)

Method	mAP (%) \uparrow	PDQ (%) \uparrow	Cls NLL \downarrow	Reg NLL \downarrow	Frame Rate (FPS) \uparrow
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 III: An evaluation with no dataset shift by testing detectors on the BDD validation dataset. Results are averaged over all seven dynamic object categories.

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

and Probability Detection Quality (PDQ) metrics introduced in Sec. III-C, as well as the Negative Log Likelihood (NLL) estimates in Sec. II-E. 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. III 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. III 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 [18]. We notice that all implemented probabilistic extensions to RetinaNet are less than 2% mAP difference compared to the deterministic baseline. On the other hand, Tab. IV 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. IV 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. III shows mAP to be correlated to the classification Negative Log Likelihood (NLL). As an example, the method that achieves the lowest mAP in Tab. III, Output Redundancy, is seen to have the highest classification NLL and as such the worst classification performance. **On the other hand, Tab. III shows mAP to have no correlation to the regression NLL.** As an example, Tab. III 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. III 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. 4, Fig. 5, and Fig. 6 show bar plots of the regression and classification NLL, calibration errors, and minimum uncertainty errors [13] 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-E1). Regarding on the uncertainty estimation in classification, Fig. 4 (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. 5 (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. 6 (left) also shows that the classification minimum uncertainty error follows the same trend as the classification

Negative Log Likelihood in Fig. 4 (left) for all evaluated probabilistic object detectors. Regarding on the uncertainty estimation in regression, the right plots from Fig. 4, Fig. 5, and Fig. 6 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 $\text{IOU} \geq 0.7$ with ground truth boxes.

Finally, looking at the **runtime** of each probabilistic object detector in the last column of Tab. III, 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. **We would like to note that unless stated otherwise, conclusions made in this section are specific to our experimental setup, where we used a single-stage probabilistic object detector that is trained on autonomous driving datasets.** We leave a more general comparative study on 3D object detection and LiDAR point cloud as an interesting future work.

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. I). 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. IV. 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 in 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 proper scoring rules such as NLL as well as calibration errors for comparing and ranking probabilistic object detectors.

2. Estimating spatial uncertainty through Loss Attenuation is essential for high quality predictive distributions in our experiments. Tab. III shows that Black Box and Output Redundancy are a have a ~50% drop in PDQ scores when compared to the rest of the implemented methods. Tab. III 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. 4, 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. 5 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. II 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 sample covariance matrix in 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 [113]. 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 spatial uncertainty estimation, which is seen in Table III 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 offers no improvement in the quality of predictive uncertainty over baseline classification from deterministic RetinaNet. Unlike the benefits offered by directly modeling the spatial uncertainty, we found little benefits from modeling the classification variance using the formulation proposed by Kendall *et al.* [35] (and introduced in Sec. II-D3). Tab. III 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. 4, 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. 5 and 6 show that Loss Attenuation does have a lower classification calibration error and a lower classifica-

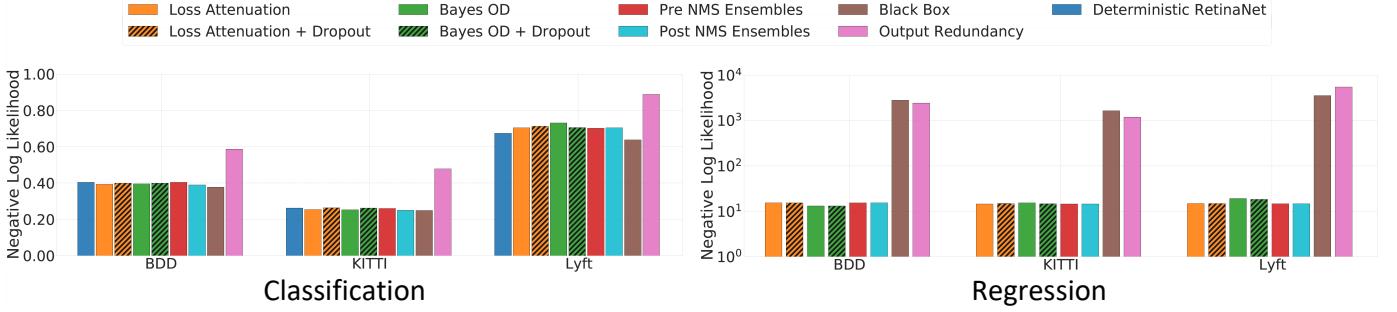


Fig. 4: Bar plots of the **Negative Log Likelihood** of the category classification and bounding box regression probabilistic output for various probabilistic detection methods implemented with the RetinaNet architecture as a backend.

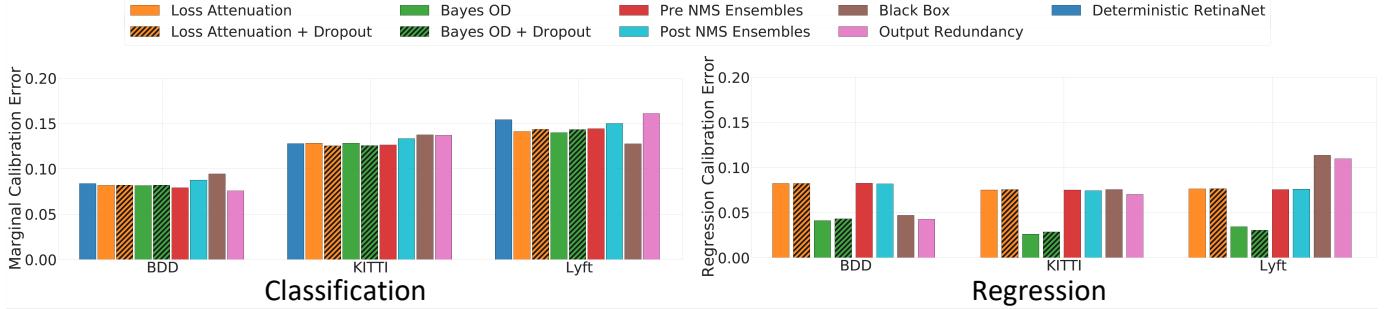


Fig. 5: Bar plots of the **calibration errors** of the category classification and bounding box regression probabilistic output for various probabilistic detection methods implemented with the RetinaNet architecture as a backend.

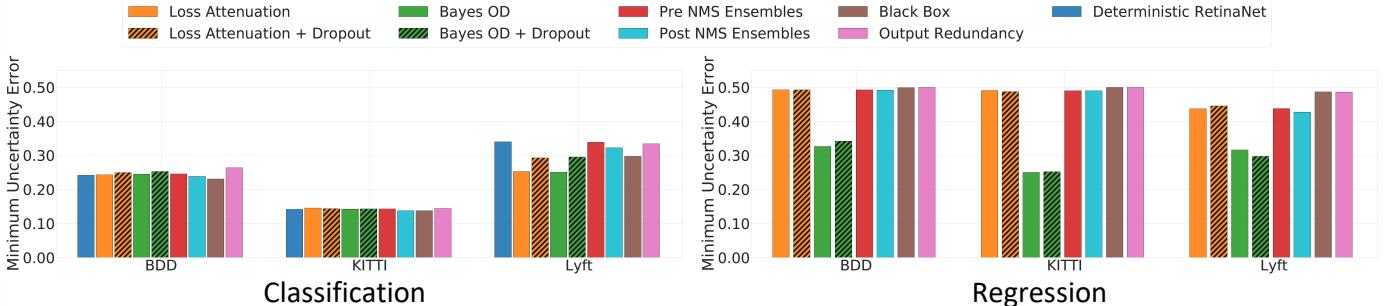


Fig. 6: Bar plots of the **minimum uncertainty errors** for the category classification and bounding box regression probabilistic output for various probabilistic detection methods implemented with the RetinaNet architecture as a backend.

tion 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 a conclusion, we find that Direct Modeling of the variance of the logit vectors does not offer improvement over learning with cross-entropy in the deterministic RetinaNet model.

4. Bayesian inference provides better spatial uncertainty qualities when compared to standard NMS. This phenomenon has been previously shown in [13], which employs Bayesian inference to fuse information from cluster members instead of selecting the detection with the highest score in the standard NMS. Tab. II 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. III 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. 5 shows that BayesOD has the lowest regression calibration error among all methods. Fig. 6 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 spatial uncertainty estimation performance comes at a cost to runtime. Table III shows that replacing NMS with Bayesian Inference in BayesOD causes a FPS from 15.21 to 9.75 when compared to Loss Attenuation, which corresponds to approx. 37ms 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

Sensors	Distance		Occlusion	
	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 V: 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.

compromised.

5. The performance gains in the quality of predictive uncertainty when using sampling-based epistemic uncertainty estimation methods does not justify their computational cost when compared to aleatoric uncertainty estimation methods. Tab. III and Tab. IV 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. III and for evaluation with dataset shift in Tab. IV. However, the difference of mAP is only within 2–3% range. Fig. 4 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. 5 and Fig. 6, respectively, where methods using ensembles exhibit similar performance to a single model trained with Loss Attenuation. On the other hand, Fig. 4, Fig. 5, and Fig. 6 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 [35], epistemic uncertainty can be explained away under large data situations such as our training on 70K 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 [64], [114]. These results from literature also translate to our results on probabilistic object detection, where Dropout enabled Loss-Attenuation and BayesOD models are shown to provide worse predictive uncertainty when compared to their Dropout-free counterparts. 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. III. We therefore conclude that the current methods used for epistemic uncertainty estimation do not provide enough improvement in the quality of estimated uncertainty to justify their high computational cost.

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

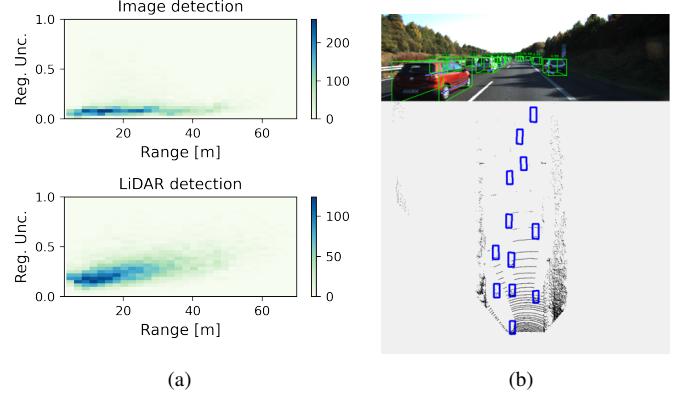


Fig. 7: (a). The heatmaps of regression aleatoric uncertainties in image detections and LiDAR detections. (b). An illustration of object detections in RGB camera images and LiDAR point clouds in the bird’s eye view.

section, we compare aleatoric uncertainty in image perception by common front-view RGB images, and LiDAR perception by Bird’s Eye View representation. To do this, we conduct experiments on the “Car” objects in the KITTI dataset [20]. 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 [11] (cf. Tab. I). Tab. V illustrates the correlations between aleatoric uncertainties in image or LiDAR perception and detection distance and occlusion, in terms of Pearson Correlation Coefficients (PCC). Defined in KITTI [20], occlusion is an ordered categorical value, with 0 indicating fully visible, 1 partly occluded, and 2 largely occluded. The aleatoric uncertainties in regression and classification are measured by the Total Variance and Shannon Entropy metrics, respectively (cf. Sec. II-E). Note that the size of bounding boxes from image detection vary significantly (big at near range and small at far range). To have a fair comparison, we normalize the Total Variance values by the diagonal length of bounding boxes (both in LiDAR and image perception). Aleatoric uncertainties (both classification and regression) in LiDAR detections are more correlated with detection distance than image detections with much larger PCC values. Fig. 7(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. 7(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. 7(b)). Note that in this experiment we only show the difference between front-view image detection and BEV LiDAR detection. Note that there exist methods which project RGB images onto Bird’s eye view [115], or LiDAR points onto front-view for 3D detection [15]. We leave a comparative study for uncertainty

modelling in these methods as an interesting future work.

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. III, which is challenging for online autonomous driving. Recently, several works attempt to estimate epistemic uncertainty without sampling. For example, Postel *et al.* [37] propose to approximate epistemic uncertainty by an error propagation method. Furthermore, Sensoy *et al.* [24] and Amini *et al.* [67] have shown that it is possible to directly predict the high-order conjugate priors of network output distributions (a Dirichlet prior for the multinomial distribution in classification [24], and a Inverse-Gamma prior for the Gaussian distribution in regression), which are used to estimate epistemic uncertainty from single-shot inference based on evidential deep learning perspective. Incorporating Error Propagation 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). Czarnecki *et al.* [59] defined seven sources of uncertainty a perception module can model during the model development and operation (cf. Appx. A). 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 [116]. Another

task that could benefit from uncertainty estimates from object detection is the object-based simultaneous localization and mapping [117], 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 [118] or for driver assistance system [119] 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. Following the work in [111], we recommend researchers to evaluate probabilistic object detection using proper scoring rules for a theoretically-sound evaluation as well as calibration errors for better insight on calibration quality of predicted probability distributions. In addition, PDQ can be useful for analyzing probabilistic object detectors, as it provides a summary of performance as a single scalar. One should note however that PDQ is **not a proper scoring rule** [111] and should not be used by itself to rank probabilistic object detectors.

Beyond proper scoring rules, 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 [120], [121] 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. [122], [123]). 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 [20] and Waymo [109]) 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 ambiguity of human annotations (which can be categorized into aleatoric uncertainty), 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, [90], [110] 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 a background knowledge of practical uncertainty estimation methods in deep learning. Next, we systematically summarize existing methods in probabilistic object detection, and evaluate uncertainty estimation methods on an one-stage 2D image detector on three standard autonomous driving datasets. Finally, we discuss challenges and open questions in this research field. 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. 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.
- [6] 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.
- [7] 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.
- [8] 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.
- [9] 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.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] 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.
- [14] 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.
- [15] 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.
- [16] G. P. Meyer and N. Thakurdesai, "Learning an uncertainty-aware object detector for autonomous driving," *arXiv preprint arXiv:1910.11375*, 2019.
- [17] 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.
- [18] 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.
- [19] 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.
- [20] 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.
- [21] 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.
- [22] D. J. C. MacKay, "A practical Bayesian framework for backpropagation networks," *Neural Computation*, vol. 4, no. 3, pp. 448–472, 1992.
- [23] R. M. Neal, "Bayesian learning for neural networks," Ph.D. dissertation, University of Toronto, 1995.
- [24] M. Sensoy, L. Kaplan, and M. Kandemir, "Evidential deep learning to quantify classification uncertainty," in *Advances in Neural Information Processing Systems*, 2018.
- [25] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural networks," in *Int. Conf. Machine Learning*, 2015.
- [26] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems*, 2011.
- [27] 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.
- [28] R. M. Neal, *Bayesian learning for neural networks*. Springer Science & Business Media, 2012, vol. 118.
- [29] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient langevin dynamics," in *Int. Conf. Machine Learning*, 2011.
- [30] T. Chen, E. Fox, and C. Guestrin, "Stochastic gradient hamiltonian monte carlo," in *Int. Conf. Machine Learning*, 2014, pp. 1683–1691.
- [31] 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.
- [32] 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.
- [33] 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.
- [34] H. Ritter, A. Botev, and D. Barber, "A scalable laplace approximation for neural networks," in *6th International Conference on Learning Representations (ICLR)*, 2018.
- [35] 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.
- [36] 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.
- [37] 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.
- [38] T. Cortinhal, G. Tzlepis, and E. E. Aksoy, "Salsanext: Fast semantic segmentation of lidar point clouds for autonomous driving," *arXiv preprint arXiv:2003.03653*, 2020.
- [39] E. Ilg, O. Çiçek, S. Gaësse, 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.
- [40] J. Gast and S. Roth, "Lightweight probabilistic deep networks," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2018.
- [41] S. Walz, T. Gruber, W. Ritter, and K. Dietmayer, "Uncertainty depth estimation with gated images for 3d reconstruction," *arXiv preprint arXiv:2003.05122*, 2020.
- [42] F. K. Gustafsson, M. Danelljan, and T. B. Schon, "Evaluating scalable bayesian deep learning methods for robust computer vision," in *CVPR Workshops*, 2020.

- [43] 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.
- [44] A. Kendall and R. Cipolla, “Modelling uncertainty in deep learning for camera relocalization,” in *IEEE Int. Conf. Robotics and Automation*, 2016.
- [45] 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.
- [46] G. Costante and M. Mancini, “Uncertainty estimation for data-driven visual odometry,” *IEEE Transactions on Robotics*, 2020.
- [47] W. H. Beluch, T. Genewein, A. Nürrnberger, 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.
- [48] 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.
- [49] 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.
- [50] K. Chitta, J. M. Alvarez, and A. Lesnikowski, “Large-scale visual active learning with deep probabilistic ensembles,” *arXiv preprint arXiv:1811.03575*, 2018.
- [51] M. Danelljan, L. V. Gool, and R. Timofte, “Probabilistic regression for visual tracking,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.
- [52] A. Hu, F. Cotter, N. Mohan, C. Gurau, and A. Kendall, “Probabilistic future prediction for video scene understanding,” *Proc. Eur. Conf. Computer Vision*, 2020.
- [53] 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.
- [54] M. Hudnell, T. Price, and J.-M. Frahm, “Robust aleatoric modeling for future vehicle localization,” in *CVPR Workshops*, 2019.
- [55] 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.
- [56] 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.
- [57] 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.
- [58] 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.
- [59] 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.
- [60] Y. Gal, “Uncertainty in deep learning,” Ph.D. dissertation, University of Cambridge, 2016.
- [61] Y. Gal, J. Hron, and A. Kendall, “Concrete dropout,” in *Advances in Neural Information Processing Systems*, 2017.
- [62] 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.
- [63] B. Lakshminarayanan, A. Pritzel, and C. Blundell, “Simple and scalable predictive uncertainty estimation using deep ensembles,” in *Advances in Neural Information Processing Systems*, 2017.
- [64] 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.
- [65] 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.
- [66] A. Malinin and M. Gales, “Predictive uncertainty estimation via prior networks,” in *Advances in Neural Information Processing Systems*, 2018.
- [67] A. Amini, W. Schwarting, A. Soleimany, and D. Rus, “Deep evidential regression,” *arXiv preprint arXiv:1910.02600*, 2019.
- [68] 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.
- [69] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, “On calibration of modern neural networks,” in *Int. Conf. Machine Learning*, 2017.
- [70] V. Kuleshov, N. Fenner, and S. Ermon, “Accurate uncertainties for deep learning using calibrated regression,” *Thirty-fifth International Conference on Machine Learning (ICML)*, 2018.
- [71] G. W. Brier, “Verification of forecasts expressed in terms of probability,” *Monthly weather review*, vol. 78, no. 1, pp. 1–3, 1950.
- [72] J. M. Bernardo, “Expected information as expected utility,” *the Annals of Statistics*, pp. 686–690, 1979.
- [73] J. Quinonero-Candela, C. E. Rasmussen, F. Sinz, O. Bousquet, and B. Schölkopf, “Evaluating predictive uncertainty challenge,” in *Machine Learning Challenges Workshop*. Springer, 2005, pp. 1–27.
- [74] J. Kohonen and J. Suomela, “Lessons learned in the challenge: making predictions and scoring them,” in *Machine Learning Challenges Workshop*. Springer, 2005, pp. 95–116.
- [75] 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.
- [76] A. Kumar, P. S. Liang, and T. Ma, “Verified uncertainty calibration,” in *Advances in Neural Information Processing Systems*, 2019.
- [77] J. Bröcker, “Reliability, sufficiency, and the decomposition of proper scores,” *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, vol. 135, no. 643, pp. 1512–1519, 2009.
- [78] T. Gneiting, F. Balabdaoui, and A. E. Raftery, “Probabilistic forecasts, calibration and sharpness,” *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 69, no. 2, pp. 243–268, 2007.
- [79] 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.
- [80] 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.
- [81] F. Kraus and K. Dietmayer, “Uncertainty estimation in one-stage object detection,” in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2019.
- [82] D. Miller, N. Sünderhauf, H. Zhang, D. Hall, and F. Dayoub, “Benchmarking sampling-based probabilistic object detectors,” in *CVPR Workshops*, 2019.
- [83] L. Bertoni, S. Kreiss, and A. Alahi, “Monoloco: Monocular 3d pedestrian localization and uncertainty estimation,” in *Proc. IEEE Conf. Computer Vision*, 2019.
- [84] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *Third International Conference on Learning Representations (ICLR)*, 2015.
- [85] 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.
- [86] 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.
- [87] 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.
- [88] 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.
- [89] 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.
- [90] 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.
- [91] X. Dong, P. Wang, P. Zhang, and L. Liu, “Probabilistic oriented object detection in automotive radar,” in *CVPR Workshops*, 2020.
- [92] 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.

- [93] 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.
- [94] 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.
- [95] Y. He, Z. Jianren Wang, Wang, and K. Jia, "Deep mixture density network for probabilistic object detection," 2020.
- [96] 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.
- [97] 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.
- [98] 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.
- [99] 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.
- [100] 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.
- [101] 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.
- [102] 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.
- [103] 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.
- [104] 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.
- [105] 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.
- [106] 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.
- [107] 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.
- [108] 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.
- [109] 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.
- [110] D. Feng, Z. Wang, Y. Zhou, W. Zhan, L. Rosenbaum, F. Timm, K. Dietmayer, and M. Tomizuka, "Labels are not perfect: Inferring spatial uncertainty in object detection," *arXiv preprint arXiv:1910.02600*, 2020.
- [111] A. Harakeh and S. L. Waslander, "Estimating and evaluating regression predictive uncertainty in deep object detectors," in *International Conference on Learning Representations*, 2021.
- [112] R. Girshick, I. Radenovic, G. Gkioxari, P. Dollár, and K. He, "Detectron," <https://github.com/facebookresearch/detectron>, 2018.
- [113] P. J. Huber, *Robust statistics*. John Wiley & Sons, 2004, vol. 523.
- [114] 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.
- [115] T. Roddick, A. Kendall, and R. Cipolla, "Orthographic feature transform for monocular 3d object detection," in *Proc. British Machine Vision Conf.*, 2019.
- [116] C. Frühwirth-Reisinger, G. Krispel, H. Possegger, and H. Bischof, "Towards data-driven multi-target tracking for autonomous driving," in *Computer Vision Winter Workshop (CVWW)*, 2020.
- [117] 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.
- [118] Y. Hu, W. Zhan, and M. Tomizuka, "Probabilistic prediction of vehicle semantic intention and motion," in *IEEE Intelligent Vehicles Symp.* IEEE, 2018.
- [119] 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.
- [120] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monroy, 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.
- [121] 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.
- [122] 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.
- [123] 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.
- [124] 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.
- [125] 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.
- [126] 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.
- [127] 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.
- [128] 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.
- [129] J. Colyar and J. Halkias, "Us highway 101 dataset," *Federal Highway Administration (FHWA), Tech. Rep. FHWA-HRT-07-030*, 2007.
- [130] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on Robot Learning*, 2017.
- [131] 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.
- [132] 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.
- [133] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *Proc. Eur. Conf. Computer Vision*, 2016.
- [134] 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.
- [135] 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.
- [136] A. Krizhevsky, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.
- [137] 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.
- [138] 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.
- [139] A. Asvadi, L. Garrote, C. Premeida, P. Peixoto, and U. J. Nunes, "DepthCN: Vehicle detection using 3d-lidar and ConvNet," in *IEEE Int. Conf. Intelligent Transp. Syst.*, 2017.
- [140] 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.

- [141] Z. Tian, C. Shen, H. Chen, and T. He, "Fcos: Fully convolutional one-stage object detection," in *Proc. IEEE Conf. Computer Vision*, 2019.
- [142] 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.
- [143] 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.
- [144] 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.
- [145] 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.
- [146] 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.
- [147] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "PVCNN: Point-voxel feature set abstraction for 3d object detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2020.
- [148] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.
- [149] 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.
- [150] J. Fei, W. Chen, P. Heidenreich, S. Wirges, and C. Stiller, "Semantic voxels: Sequential fusion for 3d pedestrian detection using lidar point cloud and semantic segmentation," in *IEEE Int. Conf. Multisensor Fusion and Integration*, 2019.
- [151] 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.
- [152] 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.
- [153] 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.
- [154] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *International journal of computer vision*, vol. 128, no. 2, pp. 261–318, 2020.



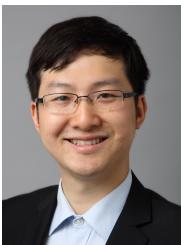
Ali Harakeh (Member, IEEE) is currently Postdoctoral Research Fellow at Mila - Quebec AI Institute. He earned his Ph.D. in 2021 at The University of Toronto Institute for Aerospace Studies (UTIAS), Toronto, Canada. During his studies, he developed real-time perception algorithms that were part of autonomous driving live demonstrations in the Consumer Electronics Show (CES) and The Vehicular Technology Conference (VTC) in 2017. He also worked on developing cutting edge perception algorithms in cooperation with companies such as Huawei Technologies, Renesas Electronics, and LG Electronics. He received his master's degree in mechanical engineering from The American University of Beirut, Beirut, Lebanon in 2016. He currently specializes in perception algorithms for autonomous driving, working on fundamental research within 2D and 3D object detection, semantic segmentation, uncertainty estimation for deep neural networks, and synthetic data generation.

Huawei Technologies, Renesas Electronics, and LG Electronics. He received his master's degree in mechanical engineering from The American University of Beirut, Beirut, Lebanon in 2016. He currently specializes in perception algorithms for autonomous driving, working on fundamental research within 2D and 3D object detection, semantic segmentation, uncertainty estimation for deep neural networks, and synthetic data generation.



Steven L. Waslander (Senior Member, IEEE) is a leading authority on autonomous aerial and ground vehicles, including multirotor drones and autonomous driving vehicles, Simultaneous Localization and Mapping (SLAM) and multi-vehicle systems. He received his B.Sc.E.in 1998 from Queen's University, his M.S. in 2002 and his Ph.D. in 2007, both from Stanford University in Aeronautics and Astronautics. He joined the University of Waterloo in 2008, where he founded and directed the Waterloo Autonomous Vehicle Laboratory (WAVELab). In

2018, he joined the University of Toronto Institute for Aerospace Studies (UTIAS), and founded the Toronto Robotics and Artificial Intelligence Laboratory (TRAILab). Prof. Waslander's innovations were recognized by the Ontario Centres of Excellence Mind to Market award for the best Industry/Academia collaboration (2012, with Aeryon Labs), as well as best paper and best poster awards at the Computer and Robot Vision Conference (2018). His work on autonomous vehicles has resulted in the Autonomoose, the first autonomous vehicle created at a Canadian University to drive on public roads.



Di Feng (Member, IEEE) received the bachelor's degree in mechatronics from the Tongji University, Shanghai, in 2014, the master's degree in electrical and computer engineering from the Technical University of Munich, Germany, in 2017, and the Dr.-Ing. degree (equivalent to Ph.D.) from the Ulm University, Germany, in 2021, in cooperation with the Corporate Research of Robert Bosch GmbH. He was also a visiting researcher at the University of California, Berkeley, in 2020/2021. Currently, he is a research engineer at the Corporate Research of Robert Bosch GmbH, where he developed perception algorithms for autonomous driving functions. Research interests include environmental perception, multi-modal object detection, uncertainty estimation in deep learning, and machine learning in robotics and autonomous driving.



Klaus Dietmayer (Member, IEEE) was born in Celle, Germany in 1962. He received his Diploma degree in 1989 in Electrical Engineering from the Technical University of Braunschweig (Germany), and the Dr.-Ing. degree (equivalent to PhD) in 1994 from the University of Armed Forces in Hamburg (Germany). In 1994 he joined the Philips Semiconductors Systems Laboratory in Hamburg, Germany as a research engineer. Since 1996 he became a manager in the field of networks and sensors for automotive applications. In 2000 he was appointed to a

professorship at the University of Ulm in the field of measurement and control. Currently he is Full Professor and Director of the Institute of Measurement, Control and Microtechnology in the school of Engineering and Computer Science at the University of Ulm. Research interests include information fusion, multi-object tracking, environment perception, situation understanding and trajectory planning for autonomous driving. Klaus Dietmayer is member of the IEEE and the German society of engineers VDI / VDE.

APPENDIX

This supplementary material to the main paper is structured as follows: Appx. A introduces epistemic and aleatoric uncertainty in details, as well as another uncertainty categorization based on the development and operation of a perception system. Appx. B explains how to estimate classification uncertainty with a distribution in softmax logits. Appx. C lists several common non-proper scoring metrics to quantify uncertainty estimates, and Appx. D discusses several uncertainty estimation benchmarks. After introducing generic object detection in Appx. E, the datasets used in our experiments are compared in Appx. F.

A. Epistemic and Aleatoric Uncertainty

In Sec. II-C, we have introduced that predictive uncertainty can be decomposed into *epistemic* uncertainty and *aleatoric* uncertainty following the Bayesian Neural Network framework. Fig. 8 shows an example of epistemic and aleatoric uncertainties in visual semantic segmentation, and Tab. VI 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 as well (cf. Sec. II-D). Furthermore, both types of uncertainty are not mutually exclusive [35]; explicitly modeling only one may result in a predictive probability that reflects the properties of the other type of uncertainty as well.

1) Another Uncertainty Categorization Method

Beyond epistemic and aleatoric uncertainty, Czarnecki *et al.* [59] 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. 9. 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).

Taking object detection as an example. Conceptual uncertainty (U1) would occur when defining class labels (e.g. shall we distinguish between “Cars” and “Vans”, or regard them as “Vehicles”?). Due to the limitation of weather conditions or locations when recording data, the training dataset could not cover all driving scenarios, which introduces U2 and U3 uncertainties. Labeling the bounding box in a RGB camera image captured during night drive is difficult due to low illumination (U4 uncertainty), and depending on the expertise of annotators, the quality of such bounding box label is different (U5 uncertainty). The network architecture has an impact on how an object detector fits the training dataset, resulting in U7 uncertainty. Finally, employing the object detector trained with highway datasets to urban scenarios would cause the domain shift problem, and result in U7 uncertainty.

B. Estimating Classification Uncertainty with Softmax Logits

Instead of the common way of using the standard softmax function with the cross-entropy loss, Kendall *et al.* [35]

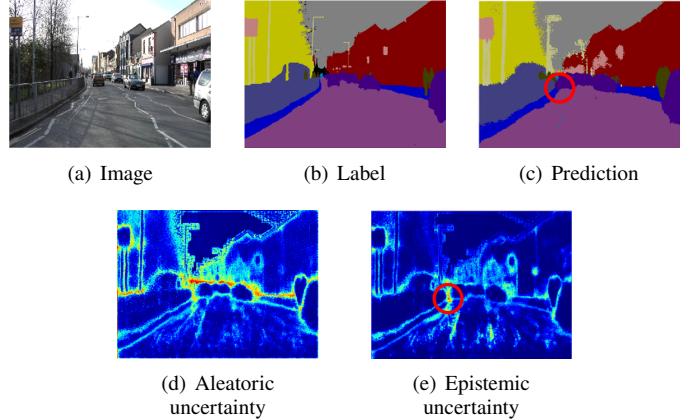


Fig. 8: An illustration of epistemic and aleatoric uncertainty in semantic segmentation on the CamVid dataset [125]. The epistemic uncertainty is measured by the Mutual Information, and the aleatoric uncertainty by the Shannon Entropy. Uncertainty estimation networks are adapted from Postel *et al.* [37]. 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 marked with circle in the image is badly-segmented, and depicts high epistemic uncertainty in (e)).

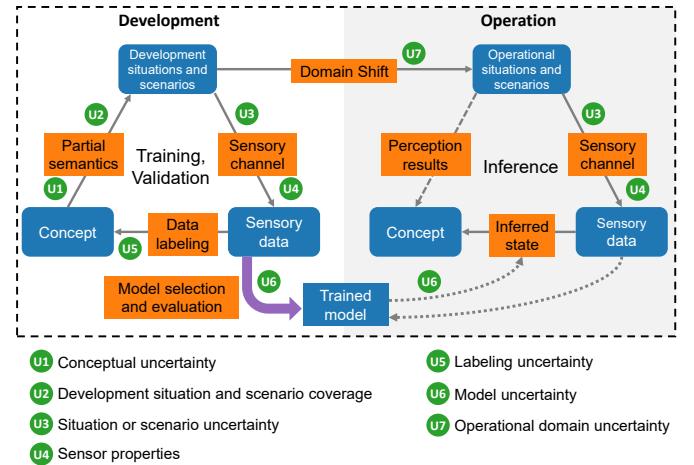


Fig. 9: Uncertainty categorization according to the procedure of developing and deploying a perception system. Seven distinct sources of uncertainty are defined. Adapted from [59].

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. A softmax logit is the unnormalized log probability output the model before the softmax function. Let a softmax logit regression variable for the class c be l_c , its distribution is given in the form: $p(l_c | \mathbf{x}, \mathbf{W}) = \mathcal{N}(l_c | \hat{\mu}_c(\mathbf{x}, \mathbf{W}), \hat{\sigma}_c^2(\mathbf{x}, \mathbf{W}))$, where $\hat{\mu}_c(\mathbf{x}, \mathbf{W}), \hat{\sigma}_c^2(\mathbf{x}, \mathbf{W})$ refer to the predicted mean and variance of

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

Problem	Example Reference	Year	Unc. Type	Dataset
Camera pose estimate	Kendall <i>et al.</i> [44]	2016	E	Cambridge Landmarks [124]
Semantic segmentation	Kendall <i>et al.</i> [36]	2017	E	CamVid [125], SUN RGB-D [126], Pascal-VOC [102]
Image annotation	Mackowiak <i>et al.</i> [48]	2018	E	Cityscapes [127]
Optical flow	Ilg <i>et al.</i> [39]	2018	E+A	KITTI [20], MPI-Sintel Flow [128]
Learning by demonstration of driving control	Choi <i>et al.</i> [65]	2018	A	US Highway 101 [129]
End-to-end driving	Michelmore <i>et al.</i> [57]	2019	E	CARLA Simulation [130]
Trajectory prediction	Hu <i>et al.</i> [52]	2020	A	Self-recorded data
Depth completion	Walz <i>et al.</i> [41]	2020	A	Gated2Depth [131]
Visual tracking	Danelljan <i>et al.</i> [51]	2020	A	OTB100 [132], UAV123 [133], NFS [134] and four others

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.* [35] 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 [19].

C. Additional Scoring Metrics For Evaluating Uncertainty In Literature

The following section introduces several common evaluation methods for uncertainty estimation. Several benchmarks based on those evaluation methods are discussed in Appx. D.

1) Shannon Entropy

Shannon Entropy (SE) is a common evaluation metric to estimate the quality of predictive uncertainty for classification tasks [60]. For a class label y of C classes, the SE, \mathcal{H} , is measured by:

$$\mathcal{H}(y|\mathbf{x}, \mathcal{D}) = - \sum_{c=1}^C p(y=c|\mathbf{x}, \mathcal{D}) \log(p(y=c|\mathbf{x}, \mathcal{D})). \quad (9)$$

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 [60], which measures the information gain of the predictive probability, when introducing the posterior distribution of model parameters \mathbf{W} :

$$\mathcal{I}(y, \mathbf{W}|\mathbf{x}, \mathcal{D}) = \mathcal{H}(y|\mathbf{x}, \mathcal{D}) - \mathbb{E}_{p(\mathbf{W}|\mathcal{D})} [\mathcal{H}(y|\mathbf{x}, \mathbf{W})], \quad (10)$$

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:

$$\mathcal{H}(y|\mathbf{x}, \mathbf{W}) = - \sum_{c=1}^C p(y=c|\mathbf{x}, \mathbf{W}) \log(p(y=c|\mathbf{x}, \mathbf{W})). \quad (11)$$

The first term on the right-hand side of Eq. 10 is calculated using Eq. 9. 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 [60] or the Deep Ensembles [63] 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) Error Curve

This method proposed by Ilg *et al.* [39] 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 with 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).

4) Total Variance (TV)

The metric proposed by Feng *et al.* [8] 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.

D. 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.* [64] conduct a large-scale study to benchmark classification uncertainty estimation, including image classification using MNIST, ImageNet [135] and CIFAR [136] datasets, as well as text and ad-click classification. They compare softmax output, MC-dropout [60], Deep Ensembles [63], temperature scaling (as the post-hoc calibration tool) [69], as well as several Bayesian inference approaches under the domain shift setting. Filos *et al.* [137] benchmark uncertainty estimation methods in the medical image application (diabetic retinopathy diagnosis). Blum *et al.* [62] build the Fishscapes dataset to benchmark uncertainty estimation methods in semantic segmentation for anomaly detection. The dataset is based on Cityscapes [127], which is a popular semantic segmentation dataset for urban autonomous driving.

E. Detailed Introduction of Generic Object Detection

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 (e.g. YOLO [138], SSD [92], and RetinaNet [19]). 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 [139] and voting [140], or by a neural network such as a Region Proposal Network (RPN) in the Faster-RCNN architecture [93]. While common object detectors require prior anchors and output post-processing stage such as Non-Maximum Suppression (NMS) to suppress duplicate detections. 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.* [141] 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.* [142] 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.

While the standard 2D convolution is the dominant method to process RGB camera images, there are several methods 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 [104], [143]–[145]. They can also be discretized in voxel [146]–[150] and processed through 3D CNN with sparse convolution operations [148], or processed in the continuous vector space without voxelization using the Point-Net based backbone [151]–[153]. For a more detailed summary we refer interested readers to [80] for object detection in 3D point clouds, and [154] for object detection in 2D images.

F. Experimental Setup: Datasets and Dataset Shift

Recent work on evaluating uncertainty estimates of deep learning models for classification tasks [64] 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** [18] 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** [20] 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** [21] 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.

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 10. Furthermore, Fig 10 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 10 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 10). The dataset

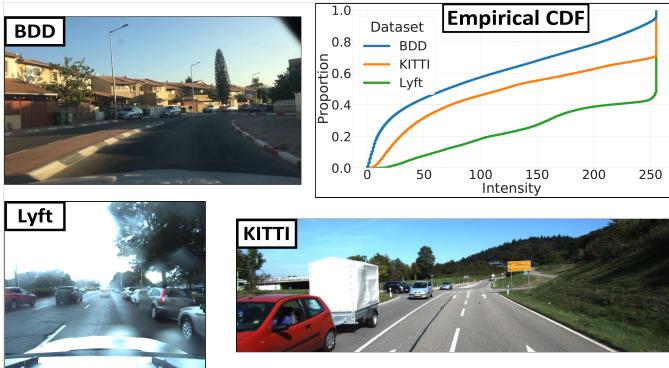


Fig. 10: Sample images from BDD, Lyft and KITTI datasets showing differences in aspect ratios, camera view points, and scene appearance between the three datasets. **Top Right:** Plot of the empirical cumulative distribution function (CDF) of the grayscale intensity values of all images in BDD, KITTI, and Lyft datasets. The CDF of KITTI is much closer to that of BDD than the CDF of Lyft.

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.