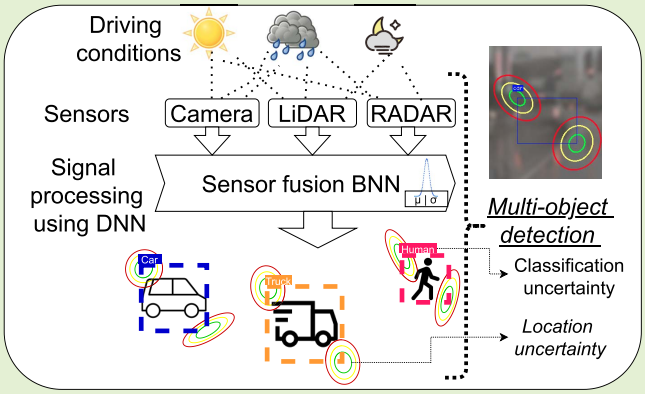
Camera, LiDAR, and Radar Sensor Fusion Based on Bayesian Neural Network (CLR-BNN)

Ratheesh Ravindran, *Member, IEEE*, Michael J. Santora, *Member, IEEE*, and Mohsin M. Jamali, *Senior Member, IEEE*

***Abstract*—*Perception* in automated vehicles (AV) is the main factor in achieving safe driving. In this perception task, multi-object detection (MOD) in diverse driving situations is the main challenge. Our recent survey [Ravindran *et al.* (2021)] shows the limitations of deep neural networks (DNN) in predicting the uncertainties of object detection in MOD. This research proposed a camera, LiDAR and RADAR sensor fusion Bayesian neural network (CLR-BNN) to improve detection accuracy and reduce uncertainties in diverse driving situations using these three primary sensing devices. The experiments were performed using the nuScence dataset with incorporation of various noises. The CLR-BNN performed better than its deterministic sensor fusion model (CLR-DNN) in terms of mAP. The CLR-BNN also showed improvement in categorical and bounding box location uncertainty using sensor fusion in diverse driving conditions. The uncertainty predictions of the CLR-BNN were validated using the calibration curve and other performance metrics.**



***Index Terms*—Sensor fusion, automated vehicles (AV), deep neural network (DNN), Bayesian neural network (BNN), camera, RADAR, LiDAR, multi-object detection (MOD).**

I. Introduction

QUALITY of perception is a crucial factor for advanced driver-assistance systems (ADAS), and this quality should increase with system maturity to address the SAE J3016^{TM} "Levels of Driving Automation." Schoettle compared human drivers and highly automated vehicles (AV) and discovered that sensing capability is critical even for fully-connected automated vehicles to be safe on the road [2]. Accuracy, robustness, reliability, and real-time performance define the quality of the perception system in AV. The camera-based (C) multi-object detection (MOD) deep neural networks (DNN) are widely studied due to their ability to classify objects, but they have limitations in detecting objects in diverse light conditions. The LiDAR (L) point cloud provides accurate spatial information about the object, the performance of LiDAR is reduced considerably during adverse environmental conditions. RADAR-based (R) object detection has improved by incorporating RADAR-target and RADAR-cube data [3]. Kutila *et al.* compared LiDAR and RADAR performance in fog and rain in a controlled chamber [4]. They proved that LiDAR's target detection is reduced considerably during rain and fog.

The MOD DNN provides a bounding box prediction, combined with a classification score. The score may tend to be over-confident, which leads to false-positive predictions [5], [6]. Such a situation is undesirable for a safety-critical system since this can cause fatal consequences. For object detection decision-making in an autonomous vehicle, it is crucial to know if there is an object and a confidence level for that object. A solution to predict object uncertainty is to have a stochastic neural network (SNN) to simulate multiple possible models [7]. Our analysis [1] shows the limitations of individual sensing modalities, the necessity to fuse heterogeneous sensor information, and limitations of current DNN techniques to predict detection uncertainties, an essential part of automated driving. A brief discussion of this analysis is presented in Section I-A.

To the best of our knowledge, the fusion of all three primary sensing devices with uncertainty estimation in diverse driving conditions has not been introduced in the AV MOD yet.

Manuscript received January 21, 2022; accepted February 15, 2022. Date of publication February 25, 2022; date of current version March 31, 2022. This work was supported in part by the University of Texas Science and Technology Acquisition and Retention (STAR) Grant. The associate editor coordinating the review of this article and approving it for publication was Dr. Priyadip Ray. *(Corresponding author: Ratheesh Ravindran.)*

Ratheesh Ravindran and Michael J. Santora are with the Electrical and Computer Engineering Department, University of Detroit Mercy, Detroit, MI 48221 USA (e-mail: ravindra@udmercy.edu).

Mohsin M. Jamali is with the Department of Electrical Engineering, University of Texas Permian Basin, Odessa, TX 79762 USA.

Digital Object Identifier 10.1109/JSEN.2022.3154980

We introduce a single-stage sensor fusion Bayesian neural network MOD, using the camera, LiDAR and RADAR (CLR-BNN). The CLR-BNN predicts the category and location bounding-box, with aleatoric uncertainties due to sensor data.

This paper's structure has six sections. The initial section of this paper contains a discussion of the current state-of-the-art in autonomous vehicle multi-object detection. The first section formulates the CLR-BNN and how the univariate and multi-variate distributions help predict the uncertainties using Bayesian inference. Also, in this section, the LiDAR and RADAR processing and sensor fusion are detailed. The details for training and testing configurations of the CLR-BNN are in the second section. In the third section, the qualitative and quantitative results for diverse driving conditions are presented. In the fourth section, the validity of the CLR-BNN predictions is discussed. A summary of the analysis is provided in the fifth section. The final section has a conclusion of this research with suggestions for future work.

A. Related Works

Recently many methods have been proposed for MOD using individual sensors [8]–[13] and sensor fusion [14]–[21]. These methods have achieved promising results on the standard open datasets with good weather conditions. A detailed analysis of various DNN's used in MOD for AV with the three primary sensors and fusion of sensors are covered in our review [1]. In this section, the state-of-the-art published works with uncertainty estimations is briefly summarized.

Stochastic neural networks (SNN) are neural networks that either use stochastic activation or stochastic weights to simulate the probabilistic distribution of model parameters. The ensemble model is one particular case of SNN [7], where predictions from multiple models are aggregated, leading to better predictions. Depending on the complexity or depth of the individual models in the ensemble, the computational resources requirement drastically increases, limiting the ensemble from computationally constrained systems, such as AV. Lakshminarayanan *et al.* proposed a deep ensemble to predict uncertainty suited for large-scale distributed computation applications [22]. The deep ensemble has M times more parameters than a single network, where M is the number of single networks in the ensemble [22], and this may limit memory-constrained applications. Due to computational constraints, Lakshminarayanan *et al.* limit evaluating the effect of deep ensembles on ImageNet (single-crop and single-class) dataset on a distributed system [22]. Tran *et al.* state that the ensemble has the highest computational training cost and overhead cost of implementation while comparing various methods for uncertainty quantification [23].

The SNN that uses Bayesian inference is referred to as Bayesian neural network (BNN) [24]. Kendall and Gal detailed the two types of predictive uncertainties, epistemic and aleatoric uncertainty [25]. The DNN model's belief after seeing the population is termed as epistemic uncertainty or model uncertainty. This epistemic uncertainty can be reduced with more training data and/or a better model. Aleatoric uncertainty, or data uncertainty, incorporates the sensor noise or ambiguities and reflects the limitation of the sensor in changing environments. The uncertainty can be predicted using the sampling (iterative forward pass for the same input) or sampling-free (single forward pass for one input) techniques [26]–[28]. The sampling-free methods are well suited for aleatoric uncertainty prediction and require less computational resources [29]. Current sampling-free techniques for 'epistemic and aleatoric uncertainty' has limitations of inapplicability of BatchNormalization [26] or significant parameter overhead [28].

Verentsov *et al.* fuse GPS and IMU signals with a probabilistic Bayesian framework to correct the trajectory of vehicles [30]. Miller *et al.* propose Monte Carlo (MC) dropout sampling for object detection for the first time, and show that uncertainty can improve object detection performance under the open-set conditions [31]. Harakeh *et al.* reformulate the standard DNN MOD inference based on camera images using MC-dropout for box co-variance regression and replaces non-maximum suppression (NMS) components with Bayesian inference [32]. The MC dropout is straightforward and has a faster training phase but lacks the expressiveness and may not fully capture the uncertainties [33]. Feng *et al.* proposed a DNN to estimate joint aleatoric and epistemic uncertainties in a 3D vehicle detector using LiDAR point-clouds [34], and shows the improvement of detection performance by modeling aleatoric uncertainty. Harakeh *et al.* and Feng *et al.* uses 10 stochastic runs of MC-Dropout, which is approximately four times slower than the sampling-free method [32].

Kraus and Dietmayer compare two-stage and single-stage MOD methods and evaluate uncertainty for a large-scale automotive pedestrian dataset using single-stage camera-based MOD [35]. Tran *et al.* extend neural network libraries to incorporate distributions over weights [36]. He *et al.* proposed a method for bounding box regression with uncertainty, using the independent single variate Gaussian distribution with the Kullback–Leibler (KL) divergence to model the localization uncertainty and improve localization performance [37]. Since He *et al.* [37] uses independent single variate Gaussian, the co-variance matrix was not taken into consideration. Feng *et al.* address uncertainties due to labeling difficulties and temporally misaligned sensors using camera and LiDAR with two-stage sensor-specific MOD. Abdar *et al.* focused on an extensive review of uncertainty quantification methods, their applications, and challenges in DNN [39].

Calibration analysis is one of the most common approaches to assess the model predictions [23]. Mitros *et al.* compare the validity of various DNN and BNN methods and prove that the DNNs suffer from poor calibration. Mitros *et al.* show that the BNN methods exhibit a lower degree of sensitivity against noisy samples than their point estimate DNN and suggest that the BNN method is the promising research direction for improving the performance. The BNN methods have better calibration than DNN methods [40]–[42].

B. Contributions

In this work a single-stage CLR-BNN for 2D MOD with aleatoric uncertainty prediction with four targets is proposed.

1) Feature and data-level fusion of three primary sensors: camera, LiDAR, and RADAR using BNN for aleatoric uncertainty.
2) Improve the MOD accuracy in terms of mean average precision (mAP), robustness, and reliability in diverse driving conditions.

3) Test the CLR-BNN performance using real-world data from the nuScense dataset [43], and artificially modified nuScenes-dataset with noisy camera-images to address various driving conditions.
4) Compare CLR-DNN (a single deterministic model as baseline) and CLR-BNN, and access CLR-BNN model predictions.

II. Designing the BNN for Sensor Fusion

The Bayesian neural network (BNN) is a DNN with distribution over the weights and biases. As a result, the BNN model outputs random variables, providing a way to measure uncertainty. In the BNN with Gaussian distributions, each weight and bias will have a mean and variance, which makes the number of parameters almost double compared to a normal DNN with similar architecture. Our analysis [1] observed that most sensor fusion DNNs use a backbone either from image-based or LiDAR point-cloud-based DNN. Jospin *et al.* address the first step for designing the BNN by selecting a DNN as a functional model [24]. This approach of selecting the backbone will enable feature extraction layer design better than starting from the basic structure for a BNN. Lin *et al.* proposed a single-stage MOD 'RetinaNet' outperforming all previous one-stage and two-stage detectors for the COCO dataset [44]. The architecture of 'RetinaNet-50' provides a balanced approach of accuracy and real-time performance [44] and is a good fit as a backbone of the 2D MOD objectives for an AV. The proposed sensor fusion BNN (CLR-BNN) architecture uses the 'RetinaNet-50' as the base with sensor fusion layers. Uni-variate and multi-variate normal distributions are used to define the weights and bias for the BNN. The parameters of distribution objects, mean (μ), variance (σ^2), and the co-variance matrix ($\sum$) are trainable. A sensor fusion deterministic network (CLR-DNN) is designed to facilitate the comparison of the CLR-BNN performance. The basic structure is similar for both CLR-DNN and CLR-BNN and is shown in Figure 1. The architecture details for both models are given in Table I.

A. Bayesian Inference

For a new input sample, x^*, the predictive distribution of a class label is given by Equation (1), where X, Y are the samples and ground truth from a dataset, and weight is represented by w.

$$P(y^*|x^*, X, Y) = \int P(y^*|x^*, w)P(w|X, Y)dw \qquad (1)$$

The main challenge for BNN is the calculation involved in the posterior probability. The $P(w|X, Y)$ is nearly impossible to calculate analytically and is intractable due to its high dimensionality and multi-modality (millions of weight parameters). This complexity increases due to the non-linear activation function between consecutive layers of DNN. There are various inference methods to solve the posterior probability, such as Markov Chain Monte Carlo (MCMC), Gibbs sampling, Hamiltonian Monte Carlo, and variational inference.

Graves proved that the Bayesian inference with the variational inference method could be made in scale (practical approach) [45], while other methods require substantial computational resources and time for computation, or the number

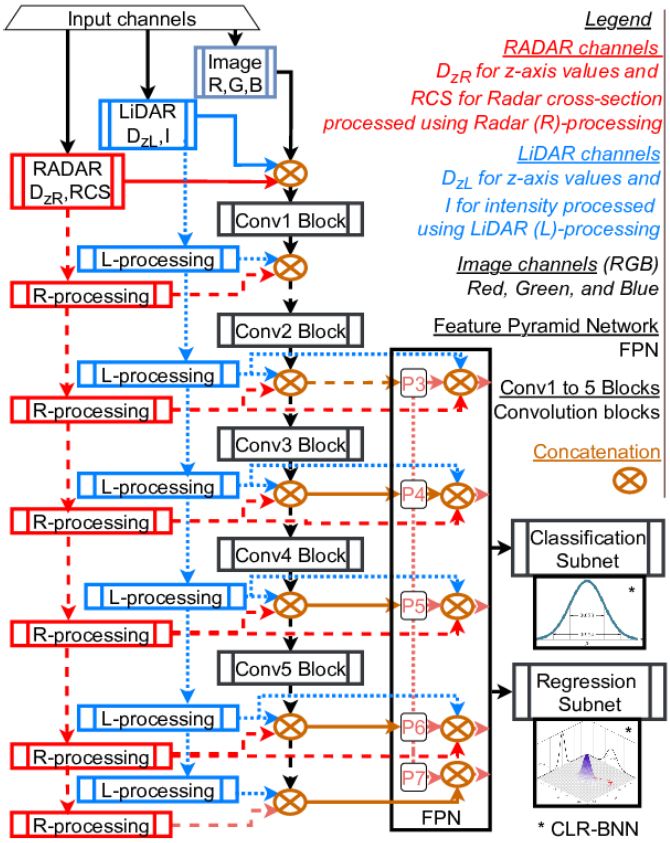


Fig. 1. CLR-DNN and CLR-BNN architecture for sensor fusion. L_processing uses the LiDAR(L) channels and R_processing uses the RADAR(R) channels. The architecture details are given in Table I. The ($\star$) represents the CLR-BNN output distributions.

of iterations is very high. The variational inference approaches this problem by approximating $P(w|X, Y)$ with more tractable distribution $q_\theta(w)$ parametrized by θ. This approximation of distribution can be performed by minimizing KL divergence or by maximizing the evidence lower bound (ELBO) [46]. This approach results in an approximated predictive distribution, as shown in Equation (2) [46].

$$q_\theta(y^*|x^*) = \int P(y^*|x^*, w)q_\theta(w)dw \qquad (2)$$

B. Layers and Initialization

One of the main requirements for the CLR-BNN is to have univariate distributions over the weights and biases for its layers, and the layers can perform convolution operations for the feature extractions. During the forward pass in training CLR-BNN, a sample from the latent distribution is used for convolution with an input observation. However, this creates a bottleneck because backpropagation cannot flow through a random node. The use of the reparameterization trick [52] addresses this bottleneck of generating the latent variable as a function of the mean (μ), standard deviation (σ), and a random noise (ϵ). This technique enables the model to backpropagate gradients and maintain the stochasticity of the latent variable through ϵ. The epistemic uncertainty calculation involves estimating the weight posterior and performing the Bayesian inference. The reparametrization trick in the CLR-BNN layers with the multiple forward passes

TABLE I
CLR-DNN AND CLR-BNN ARCHITECTURE DETAILS

	CLR-DNN	CLR-BNN
Conv1 Block (⊛)	$7 \times 7, 64, stride2$	
Conv2 Block (⊛)	$3 \times 3, maxpool, stride2$ $\begin{bmatrix} 1\times1, & 64 \\ 3\times3, & 64 \\ 1\times1, & 256 \end{bmatrix} \times 3$	
Conv3 Block (⊛)	$\begin{bmatrix} 1\times1, & 128 \\ 3\times3, & 128 \\ 1\times1, & 512 \end{bmatrix} \times 3$	
Conv4 Block (⊛)	$\begin{bmatrix} 1\times1, & 256 \\ 3\times3, & 256 \\ 1\times1, & 1024 \end{bmatrix} \times 3$	
Conv5 Block (⊛)	$\begin{bmatrix} 1\times1, & 512 \\ 3\times3, & 512 \\ 1\times1, & 2044 \end{bmatrix} \times 3$	
LiDAR(L)-Processing	Conv1 Block, Conv2 Block, Conv3 Block	
RADAR(R)-Processing	Conv1 Block, Conv2 Block	
Feature Pyramid Network (FPN) (⊛)	[feature-size] Addition and UpSampling Layers [feature size]	
Classification subnet (⊛)	[classification feature size] ×4	[classification feature size] ×4
Regression subnet (⊛)	[regression feature size] ×4. [number-of-anchors ×4]	[regression feature size] ×4 'MultivariateNormalTriL' [47] [number of anchors]×2
(⊛) Convolution from TensorFlow with ReLU activation	Conv2D	'Convolution2D Reparameterization' [48] as per the Figure 2.
Optimizer and learning rate (LR)	LAMB optimizer are used with square root LR scaling and linear-epoch warm-up to adjust learning rate [49].	

Conv blocks (1 to 5) are with BatchNormalization [50, 51].
KLDivergenceRegularizer is used for CLR-BNN.

(Monte Carlo sampling) improves the epistemic or model-based uncertainty. The aleatoric uncertainty is obtained by estimating the observation likelihood. With a single forward pass during inference, the aleatoric uncertainty is extracted in the CLR-BNN with the mean and standard deviations of the distributions. The 'Convolution2DReparameterization' layer [48] from the TensorFlow Probability (TFP) library [53] meets these requirements for CLR-BNN, and has a similar signature and operation compared to the regular convolution layers. In addition to meeting the above requirements, these CLR-BNN layers use KL divergence referring to Section II-A, and 'weights and bias' of the 'prior' and 'posterior' probabilities are defined. Selecting this TFP layer and having the 'prior' to be user-defined addresses one of the main challenges of initializing weights and bias. A pictorial representation of 'Convolution2DReparameterization' used in CLR-BNN is shown in Figure 2.

C. LiDAR Signal Processing Layers

The measurement from a LiDAR provides spatial information of the object (x, y, z) and intensity (I) of reflection. The LiDAR provides better performance in various weather conditions in comparison to the camera. The analysis [1] shows that the point-cloud projection techniques are widely used due to their lower computational requirements and are a good fit for the 2D MOD. The point-clouds are first converted from

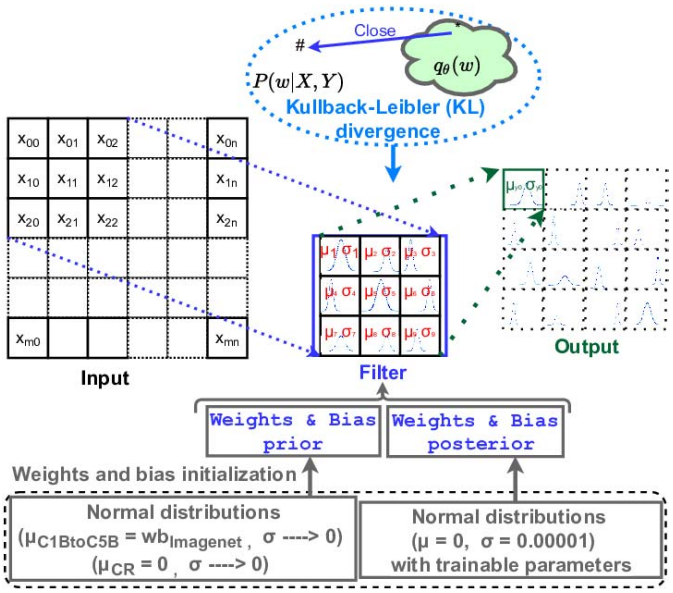


Fig. 2. 'Convolution2DReparameterization' layer used in CLR-BNN, with weights and bias initialization. $\mu_{C1BtoC5B}$ represents the mean value for feature extraction layers and μ_{CR} represents the mean value for classification and regression subnet.

TABLE II
PROCESSING METHODS FOR LiDAR AND RADAR

	Name	Processing	Risk factors
1	MaxPool2D [54]	Max pooling	may take furthermost Z value
2	MinPool2D	Min pooling	May take Zero
3	Min_IZ_Pool2D	Min pooling, ignoring Zero	
4	Conv_block	Initial blocks from 'RetinaNet-50', feature extraction layers build using 'Convolution2DReparameterization' from TFP [48].	Negatively impact sparse data as depth increases in model

vehicle coordinates to camera-view coordinates using intrinsic and extrinsic calibration matrices, which align point-clouds with the camera image. The projected LiDAR point-clouds are concatenated with the camera-image and RADAR signals for further processing as shown in Figure 3. The LiDAR point-cloud processing methods are shown in Table II. The LiDAR point-clouds are processed using 'MaxPoll2D' before processing with 'Conv_block' for the initial two stages. The LiDAR processing is represented as 'L_processing' in Figure 1. As the model depth increases the LiDAR sparse data is negatively impacted. To avoid this impact the consecutive 'L_processing' uses 'MaxPool2D' before concatenation.

D. RADAR Signal Processing Layers

A combination of RADAR-target data and RADAR-cube data [3] are the ideal data for RADAR signal processing [1]. However, this combined data is not publicly available. The electromagnetic signature of the object is the primary data from RADAR, called RADAR cross-section (RCS). The measurement from RADAR-target data provides spatial information of the object (x, y, z) and other pre-processed RADAR data, such as RCS, velocity, and others [55]. The CLR-BNN

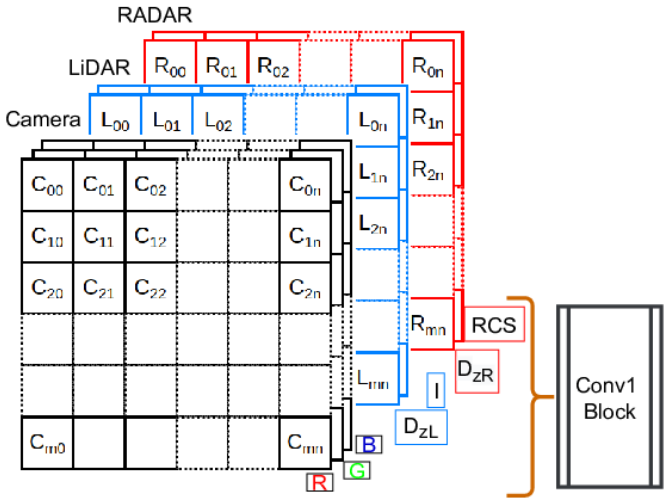


Fig. 3. Concatenation of camera image, LiDAR and RADAR data at 'Conv1 Block' level.

utilizes distance and RCS from RADAR and processes with the methods specified in Table II. Similar to LiDAR data processing, RADAR point-cloud data is projected to the camera image. The RADAR data processing is represented as 'R_processing' in Figure 1. The RADAR point-cloud is processed using 'Min_IZ_Pool2D' before processing with 'Conv_block' for the first 'R_processing' stage. Due to the negative impact of sparse RADAR data consecutive 'R_processing' functions use 'MaxPool2D' before concatenation, as shown in Figure 1.

E. Conv1 Block to Conv5 Block

These blocks are residual building blocks using identity mappings [51] as the skip connections according to 'RetinaNet-50' feature extraction layers. The CLR-BNN initializes the mean value of 'weights and bias' in 'prior' for feature extraction layers using the ImageNet represented as $wb_{Imagenet}$ in Figure 2, and the variance as 0.00001. This initialization formulates the 'prior' as a Gaussian distribution, with $\sigma^2 \rightarrow 0$, a Dirac delta function. Modeling the 'L_processing', 'R_processing' and Conv blocks (1 to 5) with univariate distributions strengthens the feature extraction and provides a way to capture small data variations.

F. Feature Pyramid Network

A Feature Pyramid Network (FPN) uses the inherent multi-scale pyramid hierarchy of deep CNNs to create feature pyramids. The pathway of building FPN is accomplished by choosing the last feature map of each group of consecutive convolution blocks that output feature maps of the same scale. The FPN layers are designed with additional stages to incorporate the LiDAR, RADAR signals with the addition to the outputs from feature extraction layers 'Conv1 Block to Conv5 Block'. The LiDAR and RADAR signals from each stage are concatenated to the FPN stage from P3 to P7 of 'RetinaNet-50-FPN' [44].

G. Classification and Regression Subnets

The 'Classification' and 'Regression' subnets are shown in Figure 1 and are detailed in Table I. The mean value of 'weights and biases' and 'variance' are initialized as zero and 0.00001, respectively, for the 'Classification and Regression subnets' initial stages, in CLR-BNN. This initialization formulates the subnets 'prior' as a Dirac delta function. The negative log-likelihood (NLL) with Focal Loss (FL) [44] is used for the 'classification subnet'. As per the ablation experiments for RetinaNet and Focal Loss (FL) in [44], the FL parameters are selected as alpha $= 0.25$, and gamma $= 2.0$. When 'Classification Subnet' output distribution with $\sigma^2 \rightarrow 0$, it means CLR-BNN is highly confident about the estimated classification. The 'Regression Subnet' are for the bounding boxes in MOD, represented as $(x1, y1, x2, y2)$. The initial stages of the 'Regression Subnet' in CLR-BNN are modeled using univariate distributions as discussed in Section II-B. The CLR-BNN uses a method with multi-variate Gaussian distribution to address the co-variance matrix for the bounding box coordinates. The final layer is modeled using 'MultivariateNormalTriL' layer [47] to incorporate the multi-variate Gaussian distribution with Cholesky decomposition, which helps co-variance matrix to be symmetric positive definite. The NLL with smooth L1 loss are used for the 'regression subnet'.

III. Experiments

A. Dataset and Experiments for CLR-BNN

A critical topic for any DNN training and testing is the selection of the dataset. There are three main criteria for selecting the dataset. The dataset is publicly available and should have all three primary sensors data with intrinsic and extrinsic calibration matrices for sensor fusion. An ideal object detector should not only perform well in good conditions but show its robustness against adverse environmental driving conditions. The nuScenes dataset [43] meets the criteria with almost 12% night and 20% rainy conditions.

The nuScence dataset is split into a training set (65%), a validation set (15%), and a test set (20%). The validation set is used for evaluation after every epochs (80 epoch and early-stopping with patience of 5) to tune the hyperparameters for controlling the learning process. Once the learning process or the training completes, the test set data and its noisy versions are used for the evaluation. The results are shared in Section IV. The validation set and test set are not used in the training process to update weights or bias. To create more diverse driving conditions an out-of-distribution dataset was created by augmenting the nuScenes dataset by adding four types of noise. The image noise, such as Impulse-Valued Noise or Salt and Pepper Noise [56], Poisson distributed random noise based on the uniqueness in the RGB values of the image channels, Speckle multiplicative noise (random noise based on the image shape) [56], and Gaussian-Blur with zero mean and 0.3 variance are used. For noisy versions of the test set for the LiDAR and RADAR signals with their respective calibration matrices relative to the camera images are used.

For training the model the LiDAR and RADAR point-clouds are filtered for the data that are inside the annotated bounding box. An example of the filtered LiDAR point-cloud data projected to the camera image is shown in Figure 4. In addition to RetinaNet's training procedure and hyperparameters [44], the CLR-BNN involves the ground truths that are modeled as Gaussian distribution with $\sigma \rightarrow 0$. As discussed in

Fig. 4. LiDAR point-clouds projected to the camera image with annotation-filter.

TABLE III
CLR-DNN EVALUATION USING TEST DATA FROM NUSCENES-DATASET (BASELINE FOR COMPARING THE CLR-BNN PERFORMANCE)

Experiments				Results			
	C	L	R	mAP ↑	Night mAP ↑	Rain mAP ↑	FPS ↑
1	✓	-	-	40.2	29.82	32.75	19.6
2	✓	✓	-	66.8	58.05	52.56	16.8
3	✓	-	✓	61.9	53.7	60.3	18.75
4	✓	✓	✓	68.85	60.75	61.7	17.4

Table legend: Camera (C), LiDAR (L), RADAR (R), mean Average Precision (mAP) and Frames Per second (FPS)

Section II-B, CLR-BNN uses 20 forward passes during the training process for each input set. The Tensorflow Mirrored-Strategy [57] with data augmentation to enable synchronous training using a single-node system with multiple NVIDIA GPUs. A TITAN XP GPU from NVIDIA is used for model performance evaluations. The test set is used for assessing the model predictions.

IV. QUALITATIVE AND QUANTITATIVE RESULTS

Qualitative results, or visualization of uncertainty in MOD, and Quantitative results using CLR-DNN and CLR-BNN are detailed in this section.

A. Quantitative Results

The CLR-DNN results are in Table III and are used as a baseline. The CLR-BNN performance evaluation experiments are performed with different sensor fusion combinations and in diverse driving conditions. Accuracy in terms of mean Average Precision (mAP), and real-time performance in frames per second (FPS) are used for the evaluations. The Tables IV to VIII summarize the performance evaluation of CLR-BNN. The effect on mAP is compared with Table III as a baseline, and the results are highlighted in respective cells of Tables IV to VIII.

B. Qualitative Results

The 'Classification Subnet' as discussed in the Section II-G outputs estimated labels with the variance. This variance is used to determine the uncertainty and is represented as 'Highly reliable', 'Reliable', and 'Unreliable' with the respective color codes on the labels as shown in Figure 6.

The 'Box Regression Subnet' as discussed in Section II-G outputs the estimated bounding box with parameters $(x1, y1, x2, y2, \sigma_{x1}^2, \sigma_{y1}^2, \sigma_{x2}^2, \sigma_{y2}^2, \sigma_{x1y1}, \sigma_{x2y2})$. The covariance matrix $(\sum)$ is represented in an elliptical form on $(x1, y1)$ and $(x2, y2)$ points of the estimated bounding box. The elliptical form at a point is calculated from the eigenvalues and eigenvectors of the covariance matrix. When variance $\rightarrow 0$ or dimensions of ellipse tends to zero, this means the output of the CLR-BNN is highly confident about the estimated

TABLE IV
CLR-BNN EVALUATION USING TEST DATA FROM NUSCENES-DATASET

Experiments				Results (Comparing with Table III in %)			
	C	L	R	mAP ↑	Night mAP ↑	Rain mAP ↑	FPS ↑
1	✓	-	-	54 (34.3)	42 (40.8)	41 (25.2)	13.1
2	✓	✓	-	75.1 (12.4)	74.2 (27.8)	62 (18.0)	11.2
3	✓	-	✓	72.9 (17.6)	71 (32.2)	71.8 (19.0)	12.5
4	✓	✓	✓	76.5 (11.1)	75 (23.5)	73 (18.3)	11.6

TABLE V
CLR-BNN EVALUATION USING TEST DATA FROM NUSCENES-DATASET WITH GAUSSIAN BLUR

Experiments				Results (Comparing with Table III in %)			
	C	L	R	mAP ↑	Night mAP ↑	Rain mAP ↑	FPS ↑
1	✓	-	-	46.5 (15.7)	32 (7.3)	33 (0.8)	13.0
2	✓	✓	-	72 (7.7)	70.2 (20.9)	59.7 (13.6)	11.0
3	✓	-	✓	70.5 (13.8)	68 (26.6)	68.8 (14.1)	12.1
4	✓	✓	✓	73.9 (7.3)	73 (20.2)	72.1 (16.9)	11.1

TABLE VI
CLR-BNN EVALUATION USING TEST DATA FROM NUSCENES-DATASET WITH SALT AND PEPPER NOISE

Experiments				Results (Comparing with Table III in %)			
	C	L	R	mAP ↑	Night mAP ↑	Rain mAP ↑	FPS ↑
1	✓	-	-	43.5 (8.2)	36 (20.7)	34 (3.8)	13.0
2	✓	✓	-	72.1 (7.9)	68.2 (17.5)	58.7 (11.7)	11.0
3	✓	-	✓	71.5 (15.4)	67.8 (26.2)	68.1 (12.9)	12.2
4	✓	✓	✓	74.1 (7.6)	72 (18.5)	71.8 (16.4)	11.4

TABLE VII
CLR-BNN EVALUATION USING TEST DATA FROM NUSCENES-DATASET WITH POISSON-DISTRIBUTED NOISE

Experiments				Results (Comparing with Table III in %)			
	C	L	R	mAP ↑	Night mAP ↑	Rain mAP ↑	FPS ↑
1	✓	-	-	45.5 (13.2)	31 (4.0)	36 (9.9)	13.0
2	✓	✓	-	71 (6.2)	62.2 (7.1)	55.7 (6.0)	11.2
3	✓	-	✓	70.5 (13.8)	58 (8.0)	65.8 (9.1)	12.3
4	✓	✓	✓	74.9 (8.8)	66 (8.6)	67.1 (8.8)	11.0

TABLE VIII
CLR-BNN EVALUATION USING TEST DATA FROM NUSCENES-DATASET WITH SPECKLE NOISE

Experiments				Results (Comparing with Table III in %)			
	C	L	R	mAP ↑	Night mAP ↑	Rain mAP ↑	FPS ↑
1	✓	-	-	49 (21.9)	38.2 (28.1)	38 (16.0)	13.4
2	✓	✓	-	73.1 (9.4)	72.2 (24.4)	58 (10.4)	11.0
3	✓	-	✓	70.9 (14.4)	68 (26.6)	69 (14.4)	12.3
4	✓	✓	✓	74.5 (8.2)	74 (21.8)	73.5 (19.1)	11.5

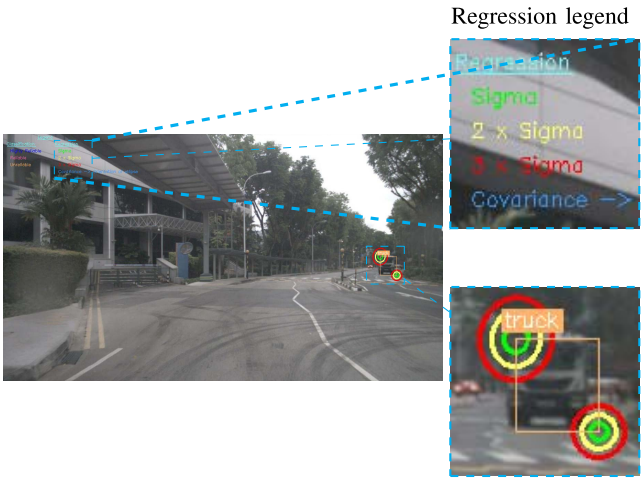


Fig. 5. MOD with camera image using CLR-BNN.

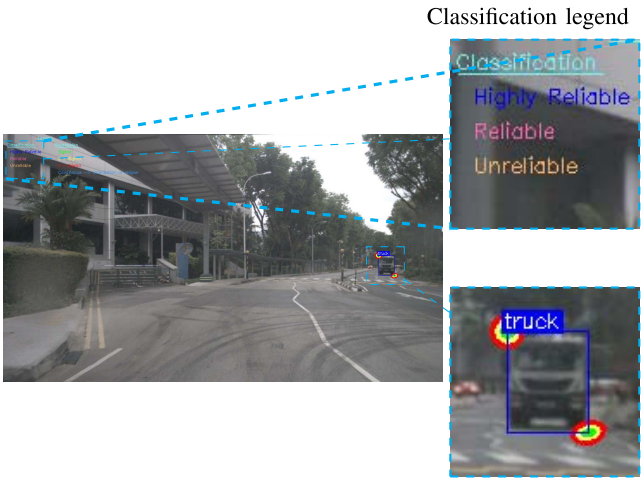


Fig. 6. MOD with camera image, LiDAR and RADAR fusion using CLR-BNN.

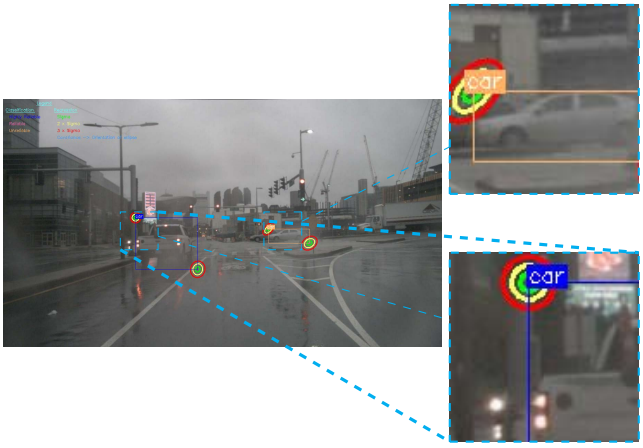


Fig. 7. MOD with camera image in rain using CLR-BNN.

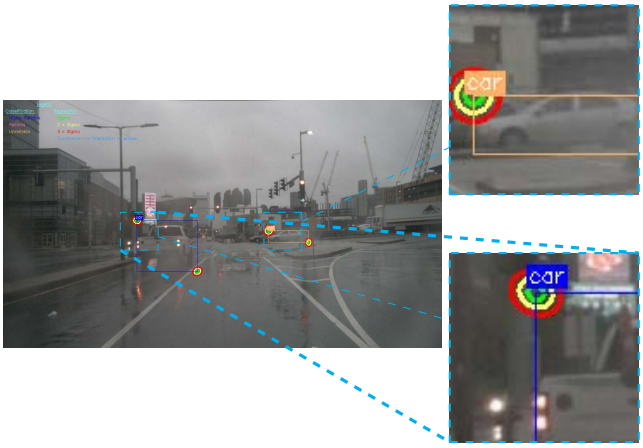


Fig. 8. MOD with camera image and LiDAR fusion in rain using CLR-BNN.

Fig. 9. MOD with camera image, LiDAR and RADAR fusion in rain using CLR-BNN.

bounding box location. When the location is estimated inaccurately due to various diverse driving conditions CLR-BNN is expected to be able to predict larger variance or larger ellipses. The ellipses are magnified for σ, 2σ, and 3σ to improve the quality of visualization, and are plotted on the image at the respective points. The projected region of interest (ROI) using the latex tool 'tikzpicture' is a five times magnification and is displayed near each figure. The qualitative results are shown in Figure 5 through Figure 13.

V. Validation of Model Predictions

Uncertainty quantification (UQ) convey information about when a model's output should or should not be trusted. Accessing the UQ with no ground truth for UQ measurements is explicitly not possible. Some of the main methods to access the UQ are calibration plots [23], [58], miscalibration area [23], sharpness [23], expected calibration error (ECE) [23], and coefficient of variation [23], [59]. A calibration curve displays the true or observed proportion in each interval relative to the predicted proportion in that interval, as stated by Tran *et al.* [23] and Kuleshov *et al.* [58]. The dispersion of the uncertainty estimate is addressed by 'coefficient of variation', which is considered as a secondary screening metric [23], [59]. The CLR-BNN predictions are validated using the uncertainty toolbox developed by Chung *et al.* for the calibration curve [60], and the other performance metrics. The results are shared in Figure 14 and Figure 15.

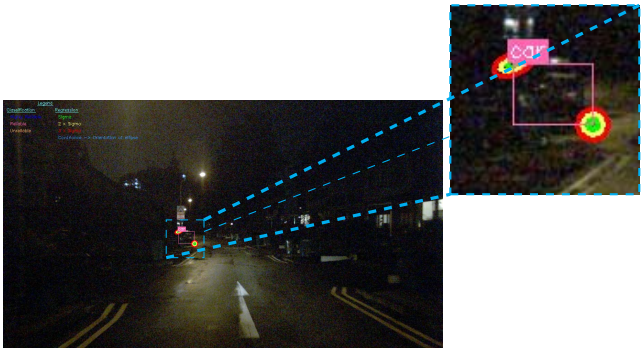


Fig. 10. MOD with camera image, LiDAR and RADAR fusion in night using CLR-BNN. This instance was not detected using camera image without fusion.

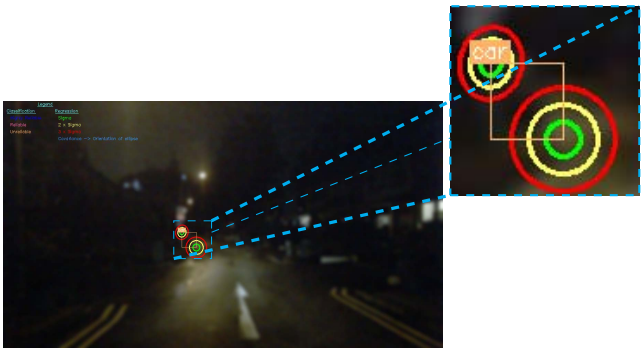


Fig. 11. MOD with camera image added with Gaussian blur noise, LiDAR and RADAR fusion in night using CLR-BNN. This instance was not detected using camera image without fusion.

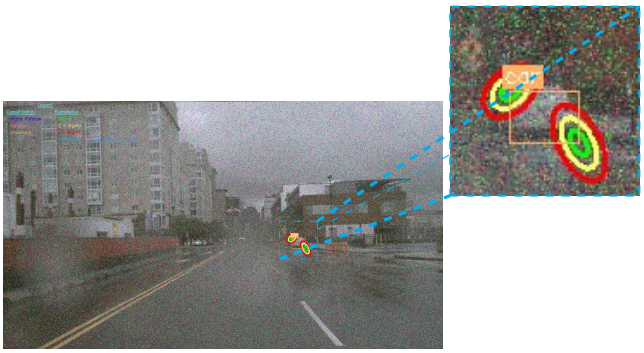


Fig. 12. MOD with camera image added with Salt and Pepper Noise, LiDAR and RADAR fusion in rain using CLR-BNN. This instance was not detected using camera image and LiDAR fusion.

VI. Summary of Analysis

Our analysis [1] motivated the work for a MOD using sensor fusion with uncertainties prediction. We propose CLR-BNN, a single-stage Bayesian neural network, to fuse the primary sensors for a MOD task with the target to improve accuracy, robustness, and reliability in diverse driving conditions. In Section II, we propose the design of the CLR-BNN using TensorFlow Probability (TFP). The advantage of using the TFP layer to incorporate the KL divergence for the Bayesian inference and addressing the challenges of initializing the weights and bias as univariate and multivariate distributions are discussed in Sections II-B and II-E to II-G.

Fig. 13. MOD with camera image added with Poisson noise, LiDAR and RADAR fusion at night using CLR-BNN. This instance was not detected using camera image and RADAR fusion.

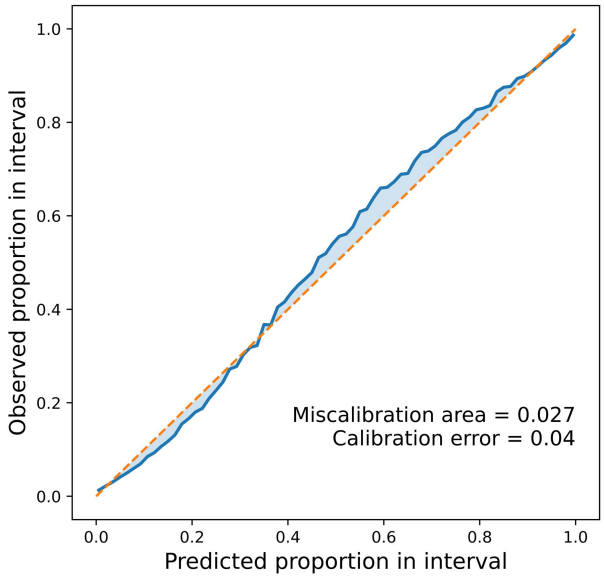


Fig. 14. Validating CLR-BNN uncertainties using calibration curve and other performance metrics.

The CLR-BNN performance evaluations are performed on a single GPU system to simulate computationally constrained AV systems. In Sections IV-A and IV-B the CLR-BNN performance are shared. The ensembling techniques are computationally constrained as discussed in Section I-A and suited for a distributed system. The performance comparison with CLR-DNN (deterministic model) shows the mAP improvements in MOD using CLR-BNN in various driving conditions***, such as*** rainy, night and artificially created conditions by adding noise to the camera images in Section IV-A. For instance, with camera image data shown in Figure 5, the object was detected as 'unreliable' marked in orange, and with elliptical radius representing the variance at (x_1, y_1) and (x_2, y_2). For the same instance, with sensor fusion shown in Figure 6, the object was detected as 'Highly reliable' marked in blue and with a reduced elliptical radius representing the reduced variance of the bounding box. The comparison of Figures 5 and 6 clearly shows the advantage of fusing the sensors by improving the category and location bounding box uncertainties. In Table IV the mAP improvements of CLR-BNN using various sensor fusion, and the effects of mAP in diverse driving conditions,

TABLE IX
CLR-BNN PERFORMANCE COMPARISON WITH 2D MOD

Experiments					Results			
	Dataset	C	L	R	mAP ↑	FPS ↑	Uncertainty measurements	Diverse driving conditions
C-DNN (Ours)	nuScenes	✓	-	-	43.2	19.6	-	✓
C-BNN (Ours)	nuScenes	✓	-	-	54	13.1	✓	✓
Probabilistic DNN [38]	nuScenes	✓	✓	-	57.74	-	✓	-
CL-DNN (Ours)	nuScenes	✓	✓	-	66.8	16.5	-	✓
CL-BNN (Ours)	nuScenes	✓	✓	-	**75.1**	11.2	✓	✓
RVNet [62]	nuScenes	✓	-	✓	56	-	-	✓
CRF-Net [63]	nuScenes	✓	-	✓	57.5	-	-	-
RANet [64]	nuScenes	✓	-	✓	69	-	-	-
BIRANet [64]	nuScenes	✓	-	✓	72.3	-	-	-
CR-DNN (Ours)	nuScenes	✓	-	✓	61.9	18.75	-	✓
CR-BNN (Ours)	nuScenes	✓	-	✓	**72.9**	12.5	✓	✓
CLR-DNN (Ours)	nuScenes	✓	✓	✓	68.85	17.4	-	✓
CLR-BNN (Ours)	nuScenes	✓	✓	✓	**76.5**	11.6	✓	✓

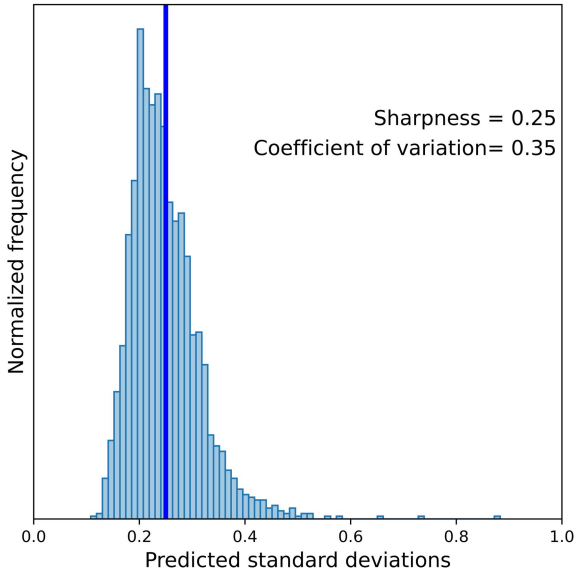


Fig. 15. Distribution plot of the CLR-BNN. Sharpness value is indicated by blue vertical line.

such as rain and night is shown. The comparison of a MOD instance in the rain, as shown in Figures 7 and 8, shows minor improvement in the location bounding box uncertainties using camera image and LiDAR fusion. Similar results can also be seen in the mAP variations in Table IV. This comparison clearly shows the limitations of LiDAR in the rain. The same instance using the fusion of all three sensors, as shown in Figure 9, highlights the improvement in categorical and in location bounding-box uncertainty prediction. The Figure 10 shows the advantage of sensor fusion during a night driving condition, whereas a camera did not detect the object in this instance.

The camera images are added with various noises to address more diverse driving conditions, and the advantage of sensor fusion and uncertainty prediction are shown in Figures 11 to 13. In an instance shown in Figure 11, the camera image is added with the Gaussian blur, this instance detection shows the categorical uncertainty is 'unreliable' and the location bounding box has a high variance. The comparison of Figures 10 and 11, and Table V shows the effect of the Gaussian blur noise. The camera-image based detection mAP has considerably reduced in the night and rainy driving conditions, as shown in Table V. The results of Figure 12 is an example of an instance that shows the effect of the camera image added with 'salt & pepper' noise. The quantitative effects of the camera image added with 'salt & pepper' noise on CLR-BNN are shared in Table VI. The Poisson-noise becomes stronger in bright areas and affects the MOD during the night. The Figure 13 and the Table VII highlight the effect of Poisson noise added to the camera image. The instance shown in Figure 13 was not detected using the camera and RADAR fusion, due to the limitation of the camera at night and the low resolution of RADAR. The Table VIII shows the analysis of speckle-noise added with the camera image. The CLR-BNN can perform better (7.3% to 11.1%) than its deterministic sensor fusion (CLR-DNN) in terms of mAP.

Real-time performance analysis in terms of frames per second (FPS) in Tables IV to VIII shows the reduction when incorporating the point-cloud from LiDAR and RADAR. This reduction due to sensor data can be improved by optimizing the CLR-BNN input data pipeline in the future. Comparing the frames per second (FPS) performance (computational cost) in Table III and Table IV, the CLR-BNN has approximately 30% difference. Sharma *et al.* proved that the pruning of BNN models could improve the real-time performance without reducing the accuracy [61]. The CLR-BNN FPS can be improved in the future by pruning methods without reducing the accuracy. Shown in the quantitative and qualitative analysis of CLR-BNN is the robustness of the system to overcome the limitations of individual sensors using sensor fusion in various driving conditions.

The CLR-BNN uncertainty predictions are validated using calibration curve and other performance metrics. The calibration could be quantified by the closeness of the calibration curve to the ideal diagonal curve. This is quantified by calculating the area between the calibration curve and the ideal diagonal, this is called as miscalibration area [23]. The Figure 14 shows validation of CLR-BNN predictions with minimal miscalibration area and the other

performance metrics. The prediction of categorical and bounding box location uncertainties provides a way to build the reliability of the MOD using CLR-BNN.

A. Performance Comparison

Performance comparison of various sensor fusion DNN was covered in our analysis [1]. This section compares the CLR-BNN results with some of the relevant state-of-the-art published work in the 2D MOD, and this comparison is summarized in Table IX. The CLR-BNN outperform on mAP considering 2D MOD using nuScenes dataset, and addressing uncertainty in various driving conditions as shown in Table IX.

VII. Conclusion and Future Perspectives

Quality in terms of accuracy, robustness, reliability, and real-time performance are the crucial and challenging factors for ADAS in AV. This article proposes CLR-BNN, a single-stage Bayesian neural network, to address an AV's MOD in various driving conditions using sensor fusion and predicting uncertainty. The CLR-BNN, developed using the probabilistic layers, allows predicting categorical and bounding box location uncertainties. The qualitative and quantitative results of various CLR-BNN experiments shows increase in mAP and decrease in FPS compared to the point estimate DNN. Pruning the CLR-BNN might improve the FPS performance without reducing the accuracy. The results also show the robustness and reliability of using sensor fusion in various driving conditions. The CLR-BNN uncertainty predictions are validated using the calibration curve, miscalibration area, sharpness, calibration error, and coefficient of variation. The validation of CLR-BNN predictions show minimal miscalibration area and estimated calibration error. The uncertainty information from sensor fusion can be used to improve the activity information of various targets in AV. Propagating the detection uncertainties to the measurement updates of the subsequent AV task of multi-object tracking will improve the tracking and prediction.

Acknowledgment

The authors would like to express their gratitude to Dr. Mohan Krishnan (Professor Emeritus, University of Detroit Mercy) and Dr. Zachariah C. Alex (Professor, Vellore Institute of Technology). They also acknowledge the support of NVIDIA Corporation with the donation of the Titan XP GPU platform used in the experimental phase of this research.

References

[1] R. Ravindran, M. J. Santora, and M. M. Jamali, "Multi-object detection and tracking, based on DNN, for autonomous vehicles: A review," *IEEE Sensors J.*, vol. 21, no. 5, pp. 5668–5677, Mar. 2021.

[2] B. Schoettle, "Sensor fusion: A comparison of sensing capabilities of human drivers and highly automated vehicles," Sustain. Worldwide Transp., Univ. Michigan, Ann Arbor, MI, USA, Tech. Rep. SWT-2017-12, Aug. 2017. [Online]. Available: http://websites.umich.edu/~umtriswt/PDF/SWT-2017-12.pdf

[3] A. Palffy, J. Dong, J. F. P. Kooij, and D. M. Gavrila, "CNN based road user detection using the 3D radar cube," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 1263–1270, Apr. 2020.

[4] M. Kutila, P. Pyykonen, H. Holzhuter, M. Colomb, and P. Duthon, "Automotive LiDAR performance verification in fog and rain," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 1695–1701.

[5] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1050–1059.

[6] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1321–1330.

[7] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*. Boca Raton, FL, USA: CRC Press, 2019.

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

[9] R. Ravindran, M. J. Santora, M. Faied, and M. Fanaei, "Traffic sign identification using deep learning," in *Proc. Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Dec. 2019, pp. 318–323.

[10] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[11] B. Yang, W. Luo, and R. Urtasun, "PIXOR: Real-time 3D object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7652–7660.

[12] R. Perez, F. Schubert, R. Rasshofer, and E. Biebl, "Single-frame vulnerable road users classification with a 77 GHz FMCW radar sensor and a convolutional neural network," in *Proc. 19th Int. Radar Symp. (IRS)*, Jun. 2018, pp. 1–10.

[13] E. Schubert, F. Meinl, M. Kunert, and W. Menzel, "High resolution automotive radar measurements of vulnerable road users–pedestrians & cyclists," in *IEEE MTT-S Int. Microw. Symp. Dig.*, Apr. 2015, pp. 1–4.

[14] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3D object detection network for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6526–6534.

[15] J. Ku, M. Mozifian, J. Lee, A. Harakeh, and S. L. Waslander, "Joint 3D proposal generation and object detection from view aggregation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 1–8.

[16] D. Xu, D. Anguelov, and A. Jain, "PointFusion: Deep sensor fusion for 3D bounding box estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 244–253.

[17] V. A. Sindagi, Y. Zhou, and O. Tuzel, "MVX-Net: Multimodal VoxelNet for 3D object detection," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 7276–7282.

[18] M. Liang, B. Yang, Y. Chen, R. Hu, and R. Urtasun, "Multi-task multi-sensor fusion for 3D object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 7337–7345.

[19] S. Chadwick, W. Maddern, and P. Newman, "Distant vehicle detection using radar and vision," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 8311–8317.

[20] A. Palffy, J. F. P. Kooij, and D. M. Gavrila, "Occlusion aware sensor fusion for early crossing pedestrian detection," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Jun. 2019, pp. 1768–1774.

[21] M. Bijelic *et al.*, "Seeing through fog without seeing fog: Deep multimodal sensor fusion in unseen adverse weather," 2019, *arXiv:1902.08913*.

[22] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," 2016, *arXiv:1612.01474*.

[23] K. Tran, W. Neiswanger, J. Yoon, Q. Zhang, E. Xing, and Z. W. Ulissi, "Methods for comparing uncertainty quantifications for material property predictions," *Mach. Learn., Sci. Technol.*, vol. 1, no. 2, May 2020, Art. no. 025006, doi: 10.1088/2632-2153/ab7e1a.

[24] L. V. Jospin, W. Buntine, F. Boussaid, H. Laga, and M. Bennamoun, "Hands-on Bayesian neural networks—A tutorial for deep learning users," 2020, *arXiv:2007.06823*.

[25] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 30, 2017, pp. 5574–5584. [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/2650d6089a6d640c5e85b2b88265dc2b-Abstract.html

[26] M. Haußmann, F. A. Hamprecht, and M. Kandemir, "Sampling-free variational inference of Bayesian neural networks by variance backpropagation," in *Proc. 35th Uncertainty Artif. Intell. Conf.*, 2020, pp. 563–573.

[27] A. Wu, S. Nowozin, E. Meeds, R. E. Turner, J. Miguel Hernández-Lobato, and A. L. Gaunt, "Deterministic variational inference for robust Bayesian neural networks," 2018, *arXiv:1810.03958*.

[28] J. Schmitt and S. Roth, "Sampling-free variational inference for neural networks with multiplicative activation noise," 2021, *arXiv:2103.08497*.

[29] M. T. Le, F. Diehl, T. Brunner, and A. Knol, "Uncertainty estimation for deep neural object detectors in safety-critical applications," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 3873–3878.

[30] S. Verentsov *et al.*, "Bayesian localization for autonomous vehicle using sensor fusion and traffic signs," in *Proc. Int. Conf. Robot. Artif. Intell.*, 2017, pp. 71–74.

[31] D. Miller, L. Nicholson, F. Dayoub, and N. Sunderhauf, "Dropout sampling for robust object detection in open-set conditions," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 3243–3249.
[32] A. Harakeh, M. Smart, and S. L. Waslander, "BayesOD: A Bayesian approach for uncertainty estimation in deep object detectors," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 87–93.
[33] A. Chan, A. Alaa, Z. Qian, and M. Van Der Schaar, "Unlabelled data improves Bayesian uncertainty calibration under covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1392–1402.
[34] D. Feng, L. Rosenbaum, and K. Dietmayer, "Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3D vehicle detection," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 3266–3273.
[35] F. Kraus and K. Dietmayer, "Uncertainty estimation in one-stage object detection," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 53–60.
[36] D. Tran, M. W. Dusenberry, M. van der Wilk, and D. Hafner, "Bayesian layers: A module for neural network uncertainty," in *Proc. NIPS*, 2019, pp. 1–13.
[37] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding box regression with uncertainty for accurate object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 2883–2892.
[38] D. Feng, Y. Cao, L. Rosenbaum, F. Timm, and K. Dietmayer, "Leveraging uncertainties for deep multi-modal object detection in autonomous driving," in *Proc. IEEE Intell. Veh. Symp. (IV)*, Oct. 2020, pp. 877–884.
[39] M. Abdar *et al.*, "A review of uncertainty quantification in deep learning: Techniques, applications and challenges," *Inf. Fusion*, vol. 76, pp. 243–297, Dec. 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253521001081
[40] J. I. Mitros and B. MacNamee, "On the validity of Bayesian neural networks for uncertainty estimation," in *Proc. 27th AIAI Irish Artif. Intell. Cogn. Sci.*, Galway, Ireland, Dec. 2019, pp. 1–16.
[41] Y. Ovadia *et al.*, "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019, pp. 13991–14002. [Online]. Available: https://proceedings.neurips.cc/paper/2019/hash/8558cb408c1d76621371888657d2eb1d-Abstract.html
[42] A. Kristiadi, M. Hein, and P. Hennig, "Being Bayesian, even just a bit, fixes overconfidence in ReLU networks," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, vol. 119, Jul. 2020, pp. 5436–5446.
[43] H. Caesar *et al.*, "NuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11621–11631.
[44] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal loss for dense object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 2, pp. 318–327, Feb. 2020.
[45] A. Graves, "Practical variational inference for neural networks," in *Advances in Neural Information Processing Systems*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Red Hook, NY, USA: Curran Associates, pp. 2348–2356.
[46] D. M. Blei. (2011). *Variational Inference*. [Online]. Available: https://www.cs.princeton.edu/courses/archive/fall11/cos597C/lectures/variational-inference-i.pdf
[47] *Tfp.layers.multivariatenormaltril: Tensorflow Probability v0.13*. Accessed: Jan. 2021. [Online]. Available: https://www.tensorflow.org/probability/api_docs/python/tfp/layers/MultivariateNormalTriL
[48] *Tfp.layers.convolution2dreparameterization: Tensorflow Probability v0.13*. Accessed: Jan. 2021. [Online]. Available: https://www.tensorflow.org/probability/api_docs/python/tfp/layers/Convolution2DReparameterization
[49] Y. You *et al.*, "Large batch optimization for deep learning: Training BERT in 76 minutes," in *Proc. 8th Int. Conf. Learn. Represent.*, Addis Ababa, Ethiopia, Apr. 2020, pp. 1–37. [Online]. Available: https://openreview.net/forum?id=Syx4wnEtvH
[50] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, vol. 37, Jul. 2015, pp. 448–456.
[51] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 630–645.
[52] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," *CoRR*, vol. abs/1312.6114, May 2014. [Online]. Available: https://arxiv.org/abs/1312.6114
[53] *Tensorflow Probability*. Accessed: Jan. 2021. [Online]. Available: https://www.tensorflow.org/probability
[54] *Tf.keras.layers.maxpool2d: Tensorflow Core V2.5.1*. Accessed: Jan. 2021. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/layers/MaxPool2D
[55] *Radar Data From a Point Cloud Data File*. Accessed: Feb. 2021. [Online]. Available: https://github.com/nutonomy/nuscenes-devkit/blob/master/python-sdk/nuscenes/utils/data_classes.py#L310
[56] A. K. Boyat and B. K. Joshi, "A review paper: Noise models in digital image processing," 2015, *arXiv:1505.03489*.
[57] *Tf.distribute.mirroredstrategy*. Accessed: Jan. 2021. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/distribute/MirroredStrategy
[58] V. Kuleshov, N. Fenner, and S. Ermon, "Accurate uncertainties for deep learning using calibrated regression," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 2796–2804.
[59] D. Levi, L. Gispan, N. Giladi, and E. Fetaya, "Evaluating and calibrating uncertainty prediction in regression tasks," 2019, *arXiv:1905.11659*.
[60] Y. Chung, I. Char, H. Guo, J. Schneider, and W. Neiswanger, "Uncertainty toolbox: An open-source library for assessing, visualizing, and improving uncertainty quantification," 2021, *arXiv:2109.10254*.
[61] H. Sharma and E. Jennings, "Bayesian neural networks at scale: A performance analysis and pruning study," *J. Supercomput.*, vol. 77, no. 4, pp. 3811–3839, Apr. 2021.
[62] V. John and S. Mita, "RVNet: Deep sensor fusion of monocular camera and radar for image-based obstacle detection in challenging environments," in *Proc. PSIVT*, 2019, pp. 361–364.
[63] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp, "A deep learning-based radar and camera sensor fusion architecture for object detection," in *Proc. Sensor Data Fusion, Trends, Solutions, Appl. (SDF)*, Oct. 2019, pp. 1–7.
[64] R. Yadav, A. Vierling, and K. Berns, "Radar + RGB fusion for robust object detection in autonomous vehicle," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2020, pp. 1986–1990.

Ratheesh Ravindran (Member, IEEE) was born in Kerala, India. He received the master's degree in sensor system technology from the Vellore Institute of Technology, Vellore, India. He is currently pursuing the Ph.D. degree in electrical and computer engineering with the University of Detroit Mercy, Detroit, MI, USA. During his master's program, he worked on real-time active vibration controls using MATLAB/Simulink and piezoelectric materials. Since 2007, he has been working on the research and development for automotive sensors and systems engineering at Bosch, Bengaluru, India, in various fields. His main interests are robotics, machine learning, signal processing, control systems, sensors, and systems engineering for automated driving and electric vehicles.

Michael J. Santora (Member, IEEE) was born in Spokane, WA, USA. He received the B.S. degree in electrical engineering from Gonzaga University, Spokane, in 2004, and the M.S. and Ph.D. degrees in electrical engineering from the University of Idaho, Moscow, ID, USA, in 2005 and 2013. He has worked as a Research and Development Controls Engineer, Spokane, from 2005 to 2012. He was a Lecturer with Eastern Washington University, Cheney, WA, USA, from 2012 to 2013; a Lecturer and a Clinical Assistant Professor with the University of Idaho in 2013 to 2016; and a Lecturer AT with Gonzaga University from 2016 to 2017. Since 2017, he has been an Assistant Professor with the University of Detroit Mercy, Detroit, MI, USA. His research interests include analysis, modeling, and control of chaotic, and complex systems.

Mohsin M. Jamali (Senior Member, IEEE) received the B.Sc.(Eng.) degree in electrical engineering from Aligarh Muslim University, Aligarh, India, in 1975, the M.S. degree in electrical engineering from the University of Saskatchewan, Canada, in 1979, and the Ph.D. degree in electrical engineering from the University of Windsor, Canada, in 1984. He is currently a Professor with the Department of Electrical Engineering, The University of Texas Permian Basin. Previously, he has served with the University of Toledo, Toledo, OH, USA, from 1984 to 2018, on various professorial ranks. He is currently developing high-performance computing laboratory and 5G testbed for spectrum sharing applications. His research include parallel processing using FPGAs, multicore, GPUs and emerging architectures for radar signal processing, machine learning algorithms, deep learning, and quantum information processing. His research has been sponsored by the U.S. Department of Education, NASA, the Office of Naval Research, Air Force, and the Department of Energy. He received the Summer Faculty Research Fellowships from AFOSR. From 2014 to 2015, he was a Fulbright-Tampere University of Technology Scholar, Tampere International Center for Signal Processing, Tampere University of Technology, Finland. He was the Chair of the IEEE VLSI Systems Applications Technical Committee and the IEEE Circuits and Systems Society. He was appointed as an IEEE Circuits and System Society's Distinguished Lecturer and a Program Lecturer from 2014 to 2015.