Probabilistic Approach for Road-Users Detection

Gledson Melotti, Weihao Lu, Pedro Conde, Dezong Zhao, *Senior Member, IEEE*, Alireza Asvadi, Nuno Gonçalves, *Member, IEEE*, and Cristiano Premebida, *Member, IEEE*

***Abstract*— Object detection in autonomous driving applications implies the detection and tracking of semantic objects that are commonly native to urban driving environments, as pedestrians and vehicles. One of the major challenges in state-of-the-art deep-learning based object detection are false positives which occur with overconfident scores. This is highly undesirable in autonomous driving and other critical robotic-perception domains because of safety concerns. This paper proposes an approach to alleviate the problem of overconfident predictions by introducing a novel probabilistic layer to deep object detection networks in testing. The suggested approach avoids the traditional Sigmoid or Softmax prediction layer which often produces overconfident predictions. It is demonstrated that the proposed technique reduces overconfidence in the false positives without degrading the performance on the true positives. The approach is validated on the 2D-KITTI objection detection through the YOLOV4 and SECOND (Lidar-based detector). The proposed approach enables interpretable probabilistic predictions without the requirement of re-training the network and therefore is very practical.**

***Index Terms*— Object Detection; Overconfident prediction; Probabilistic calibration; Multimodality; Deep learning.**

I. Introduction

REMARKABLE advances in computing hardware, sensors and machine learning techniques have contributed significantly to artificial perception for autonomous driving [1], [2], [3], [4], [5]. However, even with such progresses, artificial perception in real-world driving still meets challenges [4], [6], [7], [8]. Object detection is a key aspect of perception systems and has been gradually dominated by deep learning (DL) approaches. Generally, modern DL methods export the detection confidence as the normalized scores by the Softmax function (SM) [9] or a single value obtained from the Sigmoid function (SG) [10] without considering the overconfidence or uncertainties in the predictions (see Fig. 1). Such a lack of proper uncertainty prediction and the overconfident behaviour are undesired, because objects detected as false positives may have high score values without any level of uncertainty. It can be better understood by an example: consider six deep networks trained to classify three classes of objects namely, car, cyclist, and pedestrian. The detection confidence values for each object have been obtained through a prediction layer, such as the Softmax layer, which then normalizes the values within the interval [0, 1]. As shown in Table I, the networks show satisfactory results in terms of F-scores [11], [12] on a test set. However, what would happen when an object out of the trained classes is presented to the networks? A clue to answering this question is given by Fig. 2, where an object representing 'vegetation' class[1] has been classified with an extremist prediction (*i.e.*, value very close to one, indicating overconfident behaviour) to one of the three trained classes. Ideally, the expected value for that example would be close to 0.3, as the object does not belong to any of the three classes considered in the training. More representative cases of overconfident predictions considering out-of-training distribution examples are shown in Fig. 3, considering different classes *e.g.*, 'person-sitting', 'tree', 'pole'.

The ability to properly represent the uncertainties of predictions of an object detection system would ensure safer decision-making actions, specially in autonomous driving and robotic systems which may pose threat to people's lives [19]. In the literature, the uncertainties of a deep learning model [20], [21], [22], [23] can be obtained through the predicted values (calibration techniques) or via the network weights/loss function (regularization techniques) [24], [25], [26], [27], [28], [29], [30], [31], [32], [33], [34], [35],

TABLE I

Classification Results Using F-Score Metric by Deep Network Models

Model	Car	Cyclist	Pedestrian	Average
LeNet [13]	99.17	89.08	93.79	94.02
AlexNet [14]	99.42	91.41	96.46	95.75
Inception V3 [15]	99.68	95.05	97.67	97.46
EfficientNetB1 [16]	99.84	97.43	98.74	98.67
ViT [17]	99.46	93.56	96.37	96.46
MLP Mixer [18]	98.98	87.47	92.42	92.96

Manuscript received 15 December 2021; revised 8 December 2022; accepted 14 April 2023. Date of publication 3 May 2023; date of current version 30 August 2023. This work was supported by the Fundação para a Ciência e a Tecnologia (FCT) under Project UIDB/00048/2020. The Associate Editor for this article was A. Hegyi. *(Corresponding author: Gledson Melotti.)*

Gledson Melotti is with the Federal Institute of Espírito Santo, Electrical Technical Coordination, São Mateus, Espírito Santo 29932-540, Brazil, and also with the Institute of Systems and Robotics, Department of Electrical and Computer Engineering, University of Coimbra, Coimbra 3004-531, Portugal.

Weihao Lu and Dezong Zhao are with the James Watt School of Engineering, University of Glasgow, G12 8QQ Glasgow, U.K. (e-mail: w.lu.1@research.gla.ac.uk; Dezong.Zhao@glasgow.ac.uk).

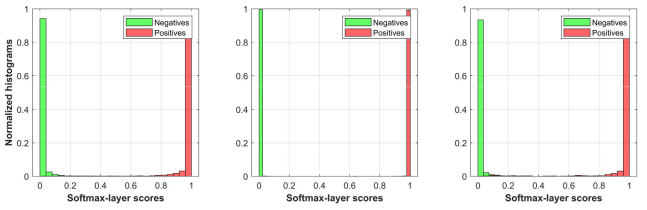
Pedro Conde and Cristiano Premebida are with the Institute of Systems and Robotics (ISR), University of Coimbra, 3004-531 Coimbra, Portugal (e-mail: cpremebida@isr.uc.pt; pedro.conde@isr.uc.pt).

Alireza Asvadi is with IADYS, 13830 Roquefort-la-Bédoule, France (e-mail: alireza.asvadi@gmail.com).

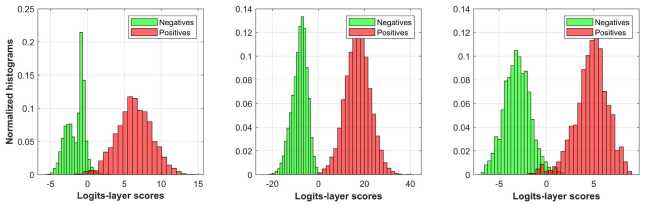
Nuno Gonçalves is with the Institute of Systems and Robotics (ISR), University of Coimbra, 3004-531 Coimbra, Portugal, and also with the Portuguese Mint and Official Printing Office (INCM), 1250-100 Lisbon, Portugal (e-mail: nunogon@deec.uc.pt).

Digital Object Identifier 10.1109/TITS.2023.3268578

[1]The vegetation class was not considered on the training set.



(a) Histogram of the Softmax-layer scores.



(b) Histogram of the logit values.

Fig. 1. In (a) we can see the overconfidence problem regarding the predictions using Softmax for a three classes case (from left to right: pedestrian, car and cyclist). The logit values (*i.e.*, the layer that feeds into Softmax) have been normalized and the corresponding distributions are shown in (b).

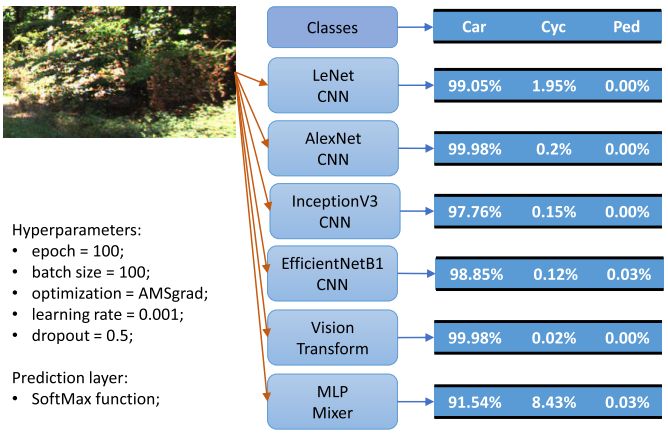


Fig. 2. Example of classifying an out-of-(training)-distribution test object. The object has been classified by six different neural networks, and all the models' outputs are overconfident - which may have critical implications.

[36], [37]. However, we will see that calibration and regularization techniques are not immune to the overconfidence problem as well, as detailed in Section II. An alternative to reduce overconfident predictions, and in some techniques to enable probabilistic interpretation, can be attained by looking at the logit-layer values (*i.e.*, the score-values before the prediction layer, or activation functions) [20], [21], [22], [23] - as illustrated in Fig. 1b which presents a more tractable distribution than the distribution out of the Softmax prediction layer.

In this context, this paper presents a new methodology to reduce overconfident predictions in deep object detection networks without interfering in the cost function and/or re-training the network. Furthermore, this paper shows that calibration techniques (such as temperature scaling and Monte Carlo Dropout, as well as confidence penalty, and Bayesian neural networks) may provide overconfidence results.

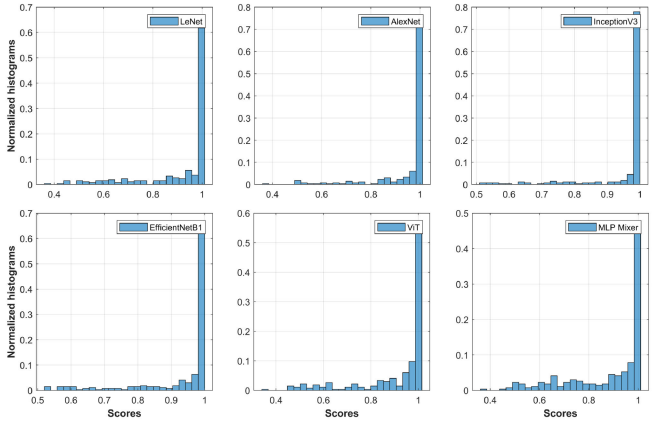


Fig. 3. Object classification on out-of-distribution test dataset through six different neural networks, using Softmax as the prediction layer, considering the LeNet [13], AlexNet [14], InceptionV3 [15], EfficientNetB1 [16], Vision Transformer [17], and MLP Mixer [18] CNNs. The overconfident behavior is notorious.

In summary, the contributions are:

- An investigation of the predicted values using distributions from the logit-layer data;
- An efficient way to obtain proper probabilistic inference via Maximum Likelihood (*ML*) and Maximum a-Posteriori (*MAP*) formulations;
- Detailed comparisons between the *ML/MAP* against the Sigmoid layer, considering true and false positive predictions by YOLOV4 and SECOND, with respect to overconfidence results;
- Comprehensive results showing that the traditional prediction layers can induce erroneous decision-making in deep object detection networks.

II. Related Work on Overconfident Predictions

Generally, the formulations that acts directly on the predicted scores to reduce overconfident predictions of learning models are considered as post-processing (or *post-hoc*) calibration techniques [31], [38], [39], [40], [41]. On the other hand, the problem of overconfident predictions in deep models can also be addressed with regularization techniques (formulations that interfere with the learning procedure of the model, to improve the generalization ability) [42], [43], [15], Bayesian models (that leverage approximate Bayesian inference instead of classical point estimation in neural networks) [36], [44], [45], or even augmentation methods [46], that produce better-calibrated models. Well-calibrated models are expected to provide accurate predictions when they are right about object detection and, conversely, provide high uncertainty when they are inaccurate about a detection. However, such techniques to reduce or mitigate overconfidence are still to be improved [25]. Actually, recent studies have shown overconfident predictions as unsolved problems in the field of deep learning [25], [47], [48], [49], [50]. Consequently, several probabilistic methods have been proposed as an alternative to reduce overconfident predictions, as well as to capture uncertainties in deep neural

network models [20], [21], [22], [23], [26], [27], [28], [29], [30], [32], [33], [34], [35], [36], [37], [44], [51], [52], [53].

The following subsections present more details about the most common and recent calibration techniques (like temperature scaling [31]), some regularization techniques (penalization of overconfident output distributions [28], [30], [32], [52], [53], label smoothing [54]) and some forms of approximate Bayesian inference (like variational inference [36] and Monte Carlo Dropout [33], [55]). Additionally, we would discuss the disadvantages of the mentioned techniques when predicting objects belonging to out-of-training-distribution data (which may be critical in autonomous driving and robotics).

A. Softmax and Sigmoid Prediction Layers

The Softmax function, a generalization of the Sigmoid function for the multiclass case, is currently one of the most commonly employed functions to act as the prediction layer in deep networks. In part, this is explained by the fact that such function increases the weights of the correct classes in an exponential way, strongly interfering in the updating of the weights, and thus may guarantee a better result in terms of classification performance. However, such behaviour may lead to overfitting, since the model becomes overconfident on the training data [56]. Additionally, the Softmax function does not provide any reliable confidence measurements for the predicted values [31], [57], [58]. Also, it is possible to find in the literature works where the Softmax's outputs are considered actual likelihood values [38], [59], [60], [61] (perhaps because they sum up to one) which tends to give an erroneous probabilistic interpretation about the results.

The Softmax, as well as the Sigmoid function, are sensitive to adversarial attacks. The studies that back this claim consider adversarial perturbations applied to the Softmax and Sigmoid prediction layer, generating possible underfitting problems on the weights [62], [63]. Additionally to the fact that Softmax and Sigmoid functions are prone to provide poorly calibrated scores and being sensitive to adversarial attacks, such functions also seem to be inadequate to cope with out-of-distribution objects in the test phase (*e.g.*, during the evaluation time the trained network can be faced with objects that do not fit to any of the training classes) as demonstrated experimentally in [21], [53], [58], [64], [65], and [66].

B. Post-Processing Calibration Techniques

Among the various existing techniques to reduce overconfident predictions, post-processing calibration techniques present the advantage of being easily applied to pre-trained models. For example, temperature scaling has demonstrated interesting characteristics because it is simple and, in some cases, efficient [31].

The value of temperature scaling (TS) is obtained by minimizing the negative log likelihood (NLL) on the validation set. All the values of the logit vector (before the prediction layer) are multiplied by a scalar parameter $\frac{1}{TS}$, with $TS > 0$. Simply, the temperature scaling parameter can be included in the Softmax prediction layer (SM)

$$SM(\hat{z_j}) = \frac{e^{(\hat{z_j}/TS)}}{\sum_{k=1}^{K} e^{(\hat{z_k}/TS)}}, \tag{1}$$

where $k \in \{1, \ldots, K\}$, K is the number of classes, $\hat{z_j}$ is the output of the predicted logit layer *i.e.*, predict score value of the object j.

C. Regularization Techniques

Different from the post-processing techniques, regularization techniques such as label smoothing and confidence penalty act during the training process, on the updates of the weights according to the cost function [15], [32], [43], [54].

For classification problems, defining $\mathbf{X} = \{\mathbf{x_1}, \ldots, \mathbf{x_j}\}$ as input data, and $\mathbf{Y} = \{\mathbf{y_1}, \ldots, \mathbf{y_j}\}$ as output data obtains the dataset $D = \{\mathbf{x_j}, \mathbf{y_j}\}_{j=1}^{N_{ts}}$, where N_{ts} is training set size, $\mathbf{x_j} \in R^n$, and $\mathbf{y_j} \in \{1, \ldots, K\}$ with K classes, the loss function considering the true label as one-hot encoding vector is defined by

$$\mathcal{L} = -\frac{1}{N_{ts}} \sum_{j}^{N_{ts}} p(\mathbf{y_j}|\mathbf{x_j}) \log(p(\hat{\mathbf{y}}_\mathbf{j}|\mathbf{x_j})), \tag{2}$$

where $p(\mathbf{y_j}|\mathbf{x_j})$ is the distribution of the true label (ground-truth) given the data, $\hat{\mathbf{y}}_\mathbf{j}$ is the predicted value for the input $\mathbf{x_j}$, and $p(\hat{\mathbf{y}}_\mathbf{j}|\mathbf{x_j})$ is the predicted labels distribution. The expression of the confidence penalty (3) includes a weighting term in the cost function given in (2). The additional term is the Entropy of the predicted values, and β is the parameter that controls the confidence penalty [32]

$$\mathcal{L} = -\frac{1}{N_{ts}} \sum_{j}^{N_{ts}} [p(\mathbf{y_j}|\mathbf{x_j}) \log(p(\hat{\mathbf{y}}_\mathbf{j}|\mathbf{x_j})) - \beta p(\hat{\mathbf{y}}_\mathbf{j}|\mathbf{x_j}) \log(p(\hat{\mathbf{y}}_\mathbf{j}|\mathbf{x_j}))]. \tag{3}$$

Unlike confidence penalty, the label smoothing technique does not interfere with the mathematical formulation of the cost function, making the model less certain about the provided predictions. In fact, label smoothing modifies the values of the one-hot encoding vector, as defined in (4) [15]

$$\mathbf{y}_{\mathbf{new_{j,k}}} = (1 - \epsilon)\mathbf{y_{j,k}} + \frac{\epsilon}{K}, \tag{4}$$

where $\mathbf{y_{j,k}}$ is the object j in the class k, $\mathbf{y}_{\mathbf{new_{j,k}}}$ is the new label value, ϵ is the smoothing parameter arbitrarily defined, and K is the number of classes. Label smoothing reduces the difference between the values of the labels of the correct class against the values of the other classes, influencing the updating of the weights of the network. Not using the label smoothing technique can cause two problems, according to [15]: "First, it may result in over-fitting: if the model learns to assign full probability to the groundtruth label for each training example, it is not guaranteed to generalize. Second, it encourages the differences between the largest logit and all others to become large, and this, combined with the bounded gradient $\frac{\partial l}{\partial z_k}$,

reduces the ability of the model to adapt. Intuitively, this happens because the model becomes too confident about its predictions".

D. Bayesian Neural Networks

Bayesian Neural Networks are modelled using approximate Bayesian inference (5) to assign probabilities to events, and thus capturing uncertainties in a model's predictions [44], [53], [45], by considering the network weights as a probability distribution parameter(s) instead of a 'deterministic' value (like in traditional deep neural networks). The posterior probability of the weights given the input and the target/class data can be expressed by [33] and [44]

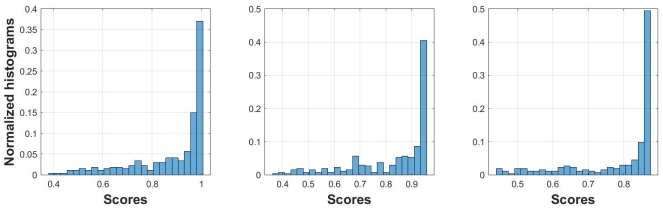
$$p(\mathbf{W}|\mathbf{X},\mathbf{Y}) = \frac{p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W})}{p(\mathbf{Y}|\mathbf{X})}, \tag{5}$$

where $\mathbf{W} = \{\mathbf{w_1}, \ldots, \mathbf{w_i}\}$ denotes the weights matrix, $\mathbf{X}$ is input data, $\mathbf{Y}$ is output data, $p(\mathbf{W})$ is the prior distribution, which expresses the uncertainty before any data observed [45], [67], and $p(\mathbf{Y}|\mathbf{X},\mathbf{W})$ is the class conditional density (likelihood function). The $p(\mathbf{Y}|\mathbf{X}) \neq 0$ acts as a scaling factor for $p(\mathbf{W}|\mathbf{X},\mathbf{Y})$, and it can be expressed as $\int p(\mathbf{Y}|\mathbf{X},\mathbf{W})p(\mathbf{W})d\mathbf{W}$ that can often be determined by the law of the total probability [45]. For example, considering a discrete case,[2] $P(\mathbf{Y}|\mathbf{X})$ can be computed per parameter $\mathbf{w_i}$ *i.e.*, $\sum P(\mathbf{Y}|\mathbf{X},\mathbf{w_i})P(\mathbf{w_i})$.

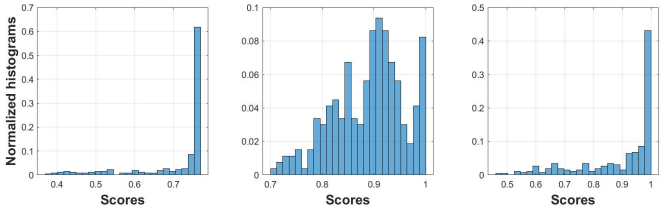
The calculation of the posterior $p(\mathbf{W}|\mathbf{X},\mathbf{Y})$ may not be trivial because the density function $p(\mathbf{Y}|\mathbf{X})$ can assume a complex form (whereas the prior can be specified from some previous knowledge and the likelihood conceivably obtained from the data). For this reason, in complex models - like deep neural networks - the posterior becomes intractable. Thus, a possible solution is to perform an approximation by means of variational inference [34], [35], [36], [52], [55], [67], [68], [45]. Nonetheless, variational inference still presents some challenges in terms of computational complexity, specially when dealing with large models and large quantities of data.

A computationally more efficient (and therefore popular) method of approximate Bayesian inference is the Monte Carlo Dropout formulation, [33], [55], that leverages *dropout* [69] (commonly used as a regularization technique) at test time, to capture the model uncertainty. Dropout [69] is a stochastic technique [66], which might potentially be included in the neural network, contributing to avoid overfitting. It is usually used during training, and therefore it can be questioned: what does occur when the dropout is used during testing? The predicted values will not be deterministic *i.e.*, the values depend on which connections between the neurons will be randomly chosen during the prediction stage. In fact, the same test sample forwarded several times in the network can have different predicted values. In [33], the authors show that applying dropout (at inference) before every weight layer of a deterministic deep neural network is equivalent to an approximation of a probabilistic deep Gaussian process.

[2]Probability formulations for continuous cases are represented by lowercase letters, while for discrete cases they are represented by uppercase letters.



(a) From left to right: temperature scaling [31], confidence penalty and label smoothing [32].



(b) From left to right: confidence penalty with label smoothing [32], Monte Carlo Dropout [55], and Bayesian neural network.

Fig. 4. Object classification on out-of-(training)-distribution test dataset using calibration and regularization techniques in an InceptionV3 CNN model.

E. Discussion on the State of the Art

Temperature scaling, confidence penalty, and label smoothing techniques aim to reduce the overconfidence problem when making predictions using relatively simple formulations. Temperature scaling also enables, as an advantage, the possibility of being applied without the need to re-train the network. The disadvantage of these techniques is the inability to directly provide an uncertainty interval regarding the detected objects subjected to the trained classes. Monte Carlo Dropout and Bayesian neural networks, on the other hand, provide uncertainties measures *i.e.*, the mean and variance associated with each confidence value, but with relatively higher computational cost.

Figure 4 shows the performance of some of the previously mentioned techniques by considering out-of-distribution test objects (person sitting, tree, pole/stem). The networks were trained from scratch to classify objects belonging to the categories {car, cyclist, pedestrian}, considering $\epsilon = 0.2$ in (4) for label smoothing, $\beta = 0.3$ in (3) for the confidence penalty, $TS = 1.82$ in (1) for temperature scaling, and for Monte Carlo Dropout the test sample was forwarded 300 times through the network. In the case of the Bayesian neural network, the classification experiments were conducted using the Tensorflow toolbox. Note that most of the objects in this controlled experiment have been classified with overconfidence.

The overconfidence problem in deep models can be detrimental to draw a firm conclusion regarding safety, particularly because it is not possible to foresee all kinds of objects that can appear, for example, within a perception system's FOV of an autonomous vehicle operating in a real-world (uncontrolled) environment. However, it can be partially concluded that the behavior shown in Fig. 4 makes it very difficult to interpret the model's confidence in a proper way.

III. Probabilistic Inference For Object Detection

This section presents a formulation to reduce overconfident predictions on existing deep object-detectors, including non-parametric and parametric modeling to represent the likelihood and the priors. The proposed approach relies on a Maximum Likelihood (ML) and Maximum a-Posteriori (MAP) function-layers, based on the Bayes rule, to replace Softmax or Sigmoid functions depending on the object detector.

A. ML and MAP Layers

The formulation behind the Bayesian inference for the proposed ML and MAP layers is built up from the logit outputs/scores (denoted by $\mathbf{x}$) and the random variables $\mathbf{C}$ and $\mathbf{W}$ *i.e.*, the class-labels and the network weight respectively. The decision layers will then output a posterior $P(\mathbf{C}|\mathbf{x}, \mathbf{W})$ that is proportional to the class-conditional density (*i.e.*, likelihood) $p(\mathbf{x}|\mathbf{C}, \mathbf{W})$ and the priors $P(\mathbf{C})$, where $\mathbf{C} = \{c_1, \ldots, c_N\}$ and $\mathbf{x} = \{x_1, \ldots, x_N\}$, with x_i corresponding to the logit value for the class c_i. Thus, the Bayes' rule may simply be given by (6), considering that the weights were the result of a learning process in order to explain the data [67] and are assumed to be constant after the training,

$$P(\mathbf{C}|\mathbf{x}) = \frac{p(\mathbf{x}|\mathbf{C})P(\mathbf{C})}{p(\mathbf{x})}. \tag{6}$$

The law of total probability [45], [70] allows (6) to be rewritten using the *per-class* discrete formulation,

$$P(c_i|\mathbf{x}) = \frac{P(\mathbf{x}|c_i)P(c_i)}{\sum_{i=1}^{K} P(\mathbf{x}|c_i)P(c_i)}, \tag{7}$$

where K is the number of classes.

Inference can then be made on the test set regarding $\mathbf{C}$ given the dependence with $\mathbf{x}$ *i.e.*, the value of the posterior probability (7) of $\mathbf{C}$ is determined after observing the value of $\mathbf{x}$. Once we have specified the likelihood distribution $p(\mathbf{x}|\mathbf{C})$, and the priors, the proposed ML/MAP prediction layers can be used to replace a Softmax or a Sigmoid function in order to output the object classification scores in a probabilistic way. Thus, the Maximum Likelihood (*ML*) and Maximum a-Posteriori (*MAP*) functions can be defined as prediction layers at the testing time, and they are expressed by

$$ML = arg\max_i \frac{(P(\mathbf{x}|c_i) + \lambda)}{\sum_{i=1}^{K}(P(\mathbf{x}|c_i) + \lambda)}, \tag{8}$$

$$MAP = arg\max_i \frac{(P(\mathbf{x}|c_i)P(c_i) + \lambda)}{\sum_{i=1}^{K}(P(\mathbf{x}|c_i)P(c_i) + \lambda)}, \tag{9}$$

where λ is an additive smoothing parameter to avoid the "zero" probability issue [71], [72], [73], to indirectly mitigate the overconfidence problem, and at the same time incorporate some level of uncertainty in the final prediction. The parameter λ is not too high or too small, and does not depend on any specific prior information, but its value has to preserve the original distribution 'shape' without degrading the final result.

Notice that, although the Bayesian formulation takes distributions into account, ML and MAP layers compute a single estimate rather than a distribution.

B. Estimating the Likelihood and Prior Probability

The non-parametric probabilistic density distribution chosen here to obtain the likelihood function comes from normalized histograms[3] of the logit-layer's scores for each class on the training dataset, as shown in Fig. 5

During the testing phase (*i.e.*, on the test set), the logit-layer score per example (or object) will then be matched to the per-class histogram, as illustrated in Fig. 5.

Unlike the likelihood function estimation, the prior probability distribution has been modelled by a Normal. Thus, the parametric estimation depends on the mean and the variance obtained from the logit scores as well (this time it is a continuous pdf as shown in Fig. 6). Therefore, the prior is $P(c_i) \sim \mathcal{N}(\mathbf{x}|\mu, \sigma^2)$ with mean μ and variance σ^2 computed per class.

The purpose of considering a discrete (normalized histogram) and a continuous pdf to model the likelihood and the *a-prior* probability respectively, is motivated from the perspective of complementary information that can be extracted from the same data.

Algorithm 1 summarizes the steps of the proposed methodology to computes *ML* and *MAP* layers scores of each class from the logit-layer values. Note that some detection models consider the objectness score (OS) parameter (parameter obtained during training), according to YOLOV4. OS is a parameter which defines whether a region in the image (grid) contains an object or not. For each grid in the image, the network provides a set of bounding-boxes, having each bounding-box an objectness score and a classification score. From an objectness threshold, the network defines which is the best bounding box that represents a given object. In other words, OS is used to evaluate which bounding box centered on a grid best represents the detected object [10]. By multiplying OS with the classification score, the resulting is the confidence level of the detected object. Thus, in the formulation of YOLOV4, the final process of defining an object's class is to multiply the objectness score with the classification score. Therefore, the proposed methodology maintains the same way of classifying an object according to the detection algorithm being analyzed. In other words, in the case of YOLOV4, the proposed methodology replaces the classification scores obtained by the Sigmoid function by the scores from the ML and MAP layers *i.e.*, multiplying the *ML* and *MAP* scores by the objectness scores.

IV. Object Detection

Currently, the state of the art in pattern recognition for autonomous driving and robotics is closely related to object detection using deep models, which has become one of the most important areas of computer vision (including LiDAR-based systems). The primary purpose of a detector is to

[3]The importance of normalizing the histogram is to ensure proper densities.

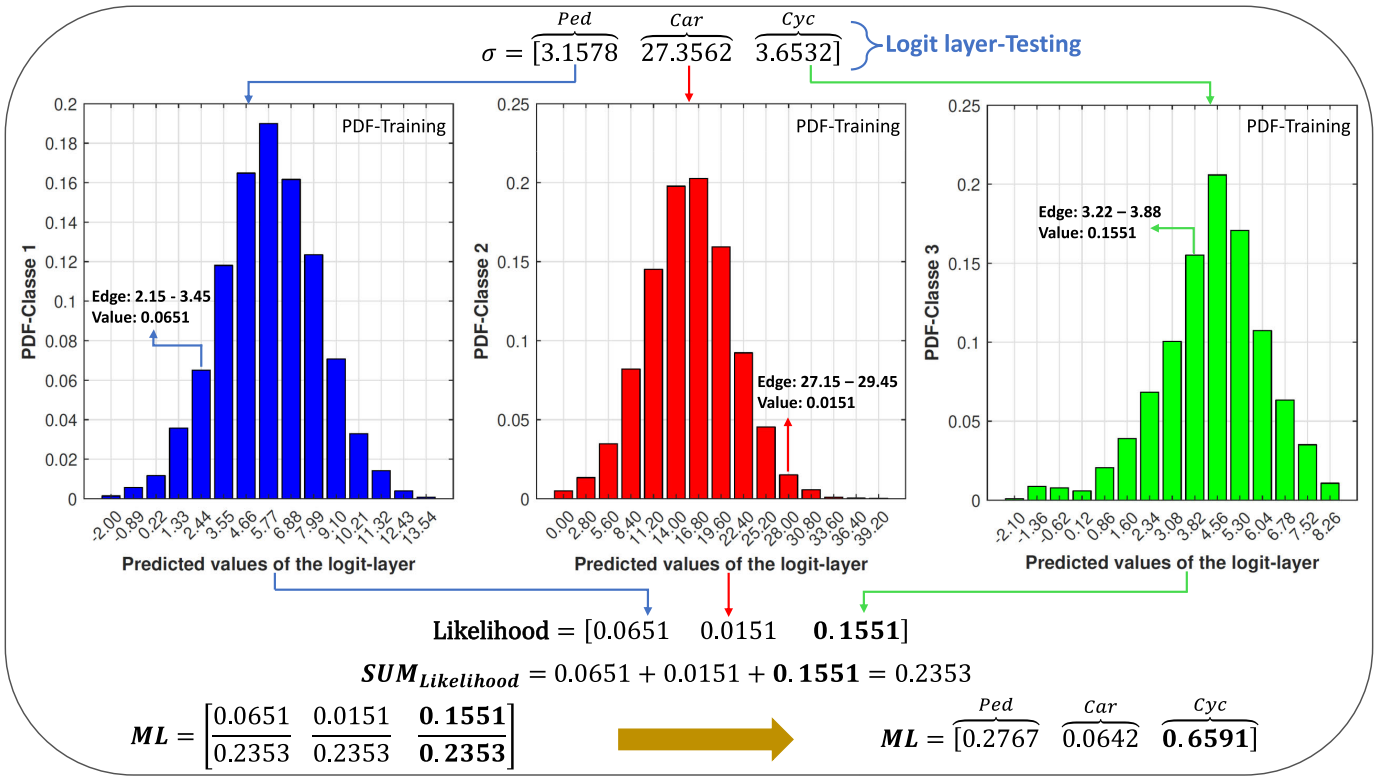


Fig. 5. Getting the probability values from normalized-histograms used to model the distributions of the logits on the training set. .

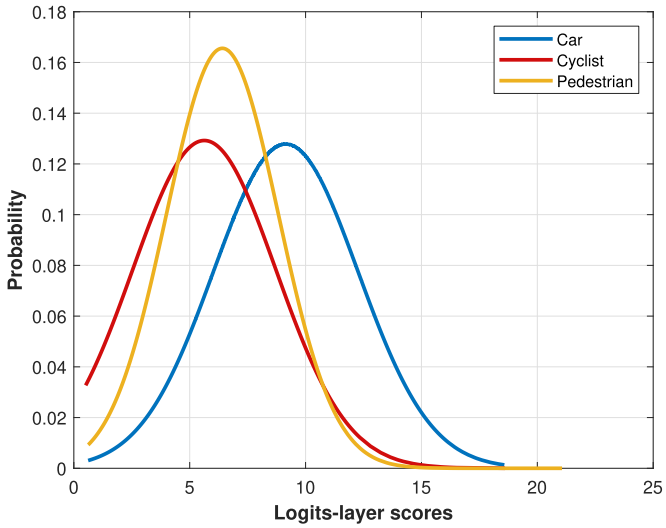


Fig. 6. Gaussian distributions to estimate the prior probabilities for the three training classes (car, cyclist and pedestrian).

estimate the object's position, size and class/category. A $2D$ detector estimates bounding boxes considering the coordinates of the center, width and height of the objects' hypothesis. Additionally, detectors estimate the classification score and predicted class. In plain words, the recent detectors rely on a series of steps to define the bounding boxes and the classification scores depending on comparisons across thresholds between predicted output and ground-truth (training stage), as well as objectness score threshold, intersection over union (IoU), non-max suppression (NMS), and class threshold.

Among the various detection models, we have chosen the YOLOV4 [10], which at the time has reached the state of the art performance on the COCO dataset, while achieving shot inference time. The structure of YOLOV4 and the proposed methodology is illustrated in Fig. 7.

The advantages of YOLOV4, over previous versions and other existing object detection algorithms, are that YOLOV4 tries to avoid overconfident results by using data augmentation (CutMix and Mosaic), class label smoothing, and dropout in the convolution layers (DropBlock regularization), which then influence the classifier accuracy. Also, unlike many object detection algorithms, YOLOV4 uses the Mish activation function instead of the traditional functions (*e.g.*, ReLU, ELU, SeLU, PReLU, Swich). Additionally, the cost function of YOLOV4 incorporates overlap area, central point distance and aspect ratio [74], as well as cosine annealing scheduler (learning rate) [75], a modified cross-iteration batch normalization [76], self-adversarial training [10]. Finally, the Sigmoid function is employed to get the final bounding boxes and the respective classification scores.

Even though YOLOV4 considers strategies to reduce overconfident predictions, our results demonstrate that a significant number of false positives are predicted with high score values, which demonstrates that the prediction layer using the Sigmoid function did not mitigate overconfident results enough, as shown in Fig. 8.

For object detection with $3D$ point clouds, we choose the lightweight yet effective SECOND [77] detector as the baseline. SECOND extracts features by encoding voxel-based $3D$ data with submanifold sparse $3D$ convolution layers [77].

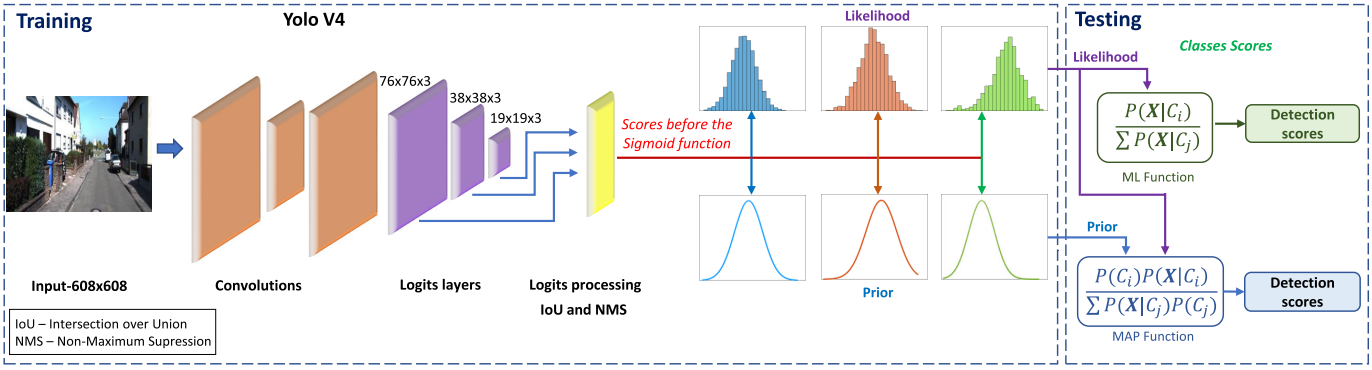


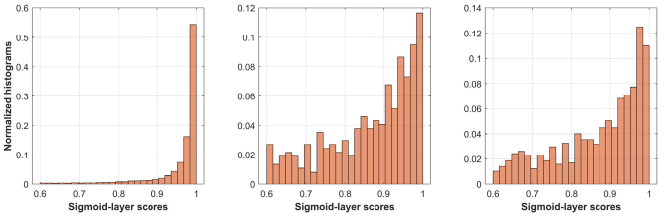
Fig. 7. YOLOV4 representation with logit and Sigmoid (*SG*) layers, Maximum Likelihood (*ML*) and Maximum a-Posterior (*MAP*) functions. After training, the predicted values from the Sigmoid Layer were replaced by the scores from *ML* and *MAP* functions. Notice that the YOLOV4 was not trained or re-trained with the *ML/MAP* functions.

Algorithm 1 *ML and MAP Layers*

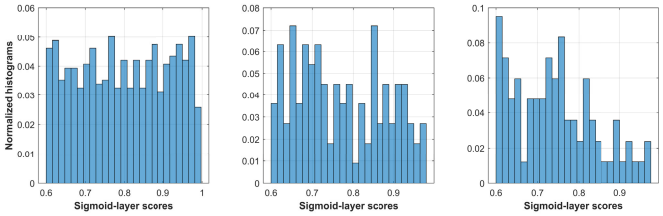
Input:
- Densities (normalized histogram and Gaussian distribution on the training set - logit-layer values, Fig. 5);
- Logit-layer values on the test set ($Test$);
- Additive smoothing (λ);
- Number of classes (K).

Output:
- Maximum Likelihood (*ML*);
- Maximum a-Posteriori (*MAP*).

Normalized frequency histograms:
$hc \leftarrow histogram(Train(\mathrm{K}))$;
Edge values of each bin of each histogram:
$BinLow \leftarrow BinEdgesLow(hc)$;
$BinHigh \leftarrow BinEdgesHigh(hc)$;
Frequency values of each of the histograms:
$V \leftarrow Values(hc)$
Getting the likelihood:
$P(\mathbf{x}|\mathbf{C}) \leftarrow zeros(size(Test), K)$;
for $k \leftarrow 1 : size(Test)$ **do**
  for $cl \leftarrow 1 : K$ **do**
    for $i \leftarrow 1 : size(BinValues)$ **do**
      if $(BinLow(cl, i) \leqslant Test(k, cl)) \,\&\, (Test(k, cl) < BinHigh(cl, i))$ **then**
        $P(\mathbf{x}|C)(k, cl) \leftarrow V(cl, i)$;
      end
    end
  end
Getting the Prior:
$P(\mathbf{C}) \leftarrow \mathcal{N}(Test|[\mu_{Train}, \sigma^2_{Train}])$;
Calculating the *ML* and *MAP*:
$ML \leftarrow P(\mathbf{x}|\mathbf{C}) + \lambda$;
$ML \leftarrow (ML/\mathrm{sum}(ML)) * \mathrm{ObjectnessScore}$;
$MAP \leftarrow P(\mathbf{x}|\mathbf{C})P(\mathbf{C}) + \lambda$;
$MAP \leftarrow (MAP/\mathrm{sum}(MAP)) * \mathrm{ObjectnessScore}$;



(a) Score distributions of the true positive objects.



(b) Score distributions of the false positive objects.

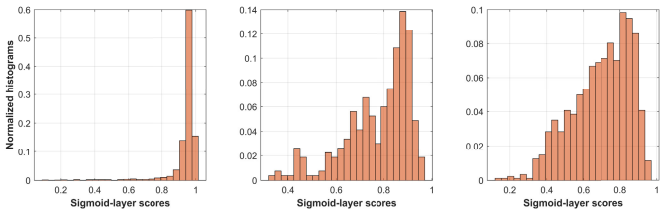
Fig. 8. Distributions of the YOLOV4's classification scores for car, cyclist, and pedestrian classes, considering RGB modality.

The $3D$ features are converted to Bird's Eye View (BEV) representations via high compression, where the height in the metric space is flattened into the feature channels. Standard $2D$ convolutions are used to generate BEV features. The outputting feature map is passed to the single-stage anchor-based detector head for classification and bounding box regression. Compared to the sophisticated models with more structure information, the voxel-based SECOND [77] has a much faster runtime with comparable performance.

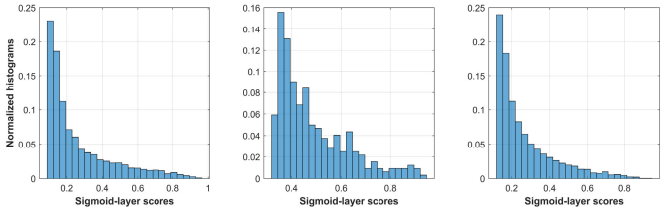
As shown in Fig. 9, SECOND [77] outputs a similar distribution, in a *lato sensu* perspective, of the true positives as YOLOV4, while giving distinct and more desirable decisions on the false positives.

A. RGB and LiDAR Modalities

The proposed probabilistic methodology is validated through multi-sensory $2D$ and $3D$ object detection on the KITTI dataset, considering for YOLOV4 detector RGB images, range-view (RaV), and reflectance-view (ReV) maps modalities, as showed in Fig. 10, and $3D$ point clouds for SECOND detector. The modalities (RaV), and (ReV) were obtained by projecting the $3D - LiDAR$ point clouds in the $2D$ image plane followed by an upsampling step using



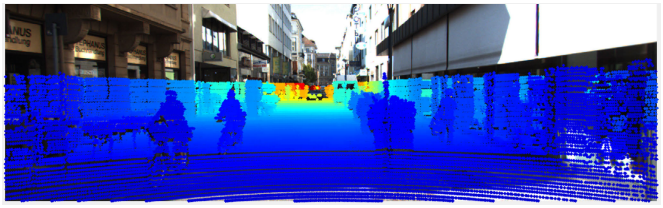
(a) Score distributions of the true positive objects.



(b) Score distributions of the false positive objects.

Fig. 9. Distributions of the SECOND's classification scores for car, cyclist, and pedestrian classes, considering LiDAR modality (3D LiDAR).

(a) RGB modality.



(b) Projection of the 3D point clouds in the 2D image plain.

Fig. 10. The 3D cloud points were obtained from the Velodyne 64 sensor and then projected onto the image plane.

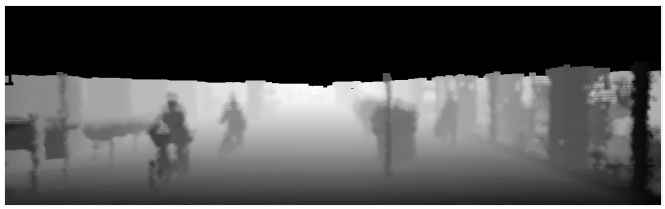
a tailored bilateral filter implementation, expressed in (10), where $\hat{r_0}$ is the upsampled pixel [78]

$$\hat{r_0} = \frac{1}{W}\sum_{i=1}^{n} G_{\sigma_s}(||c_0 - c_i||)G_{\sigma_r}(|r_0 - r_i|)r_i, \quad (10)$$

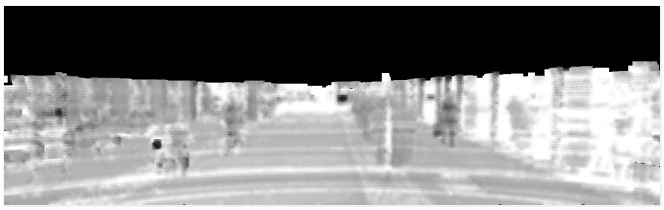
where $W = \sum_{i=1}^{n} G_{\sigma_s}(||c_0 - c_i||)G_{\sigma_r}(r_0 - r_i)$ is a scaling factor that ensures the output sums to one, G_{σ_s} weights the point c_i inversely proportional to a distance (we used the Euclidean distance), and G_{σ_r} weights the sampled points from their intensity values r_i. G_{σ_s} and G_{σ_r} were considered to be of the form

$$G_{\sigma_s} = \frac{1}{1 + (||c_0 - c_i||)}, \quad (11)$$

$$G_{\sigma_r} = \frac{1}{1 + (|r_0 - r_i|)}. \quad (12)$$



(a) RaV map generated from LiDAR's depth data.



(b) ReV map using the LiDAR's reflectance data.

Fig. 11. Maps generated by bilateral filtering using sliding window with size 13×13.

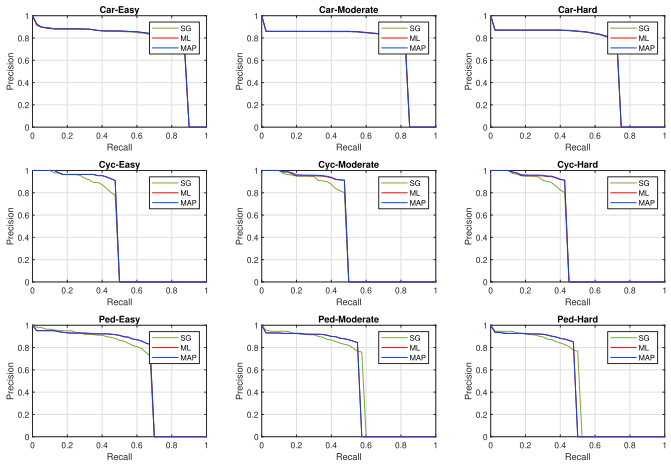


Fig. 12. Precision-recall curves for car, cyc. and ped. classes using the RGB modality, with $\lambda_{ML} = 1.6 \times 10^{-6}$, $Bins_{ML} = 22$, $\lambda_{MAP} = 1.0 \times 10^{-8}$, and $Bins_{MAP} = 24$.

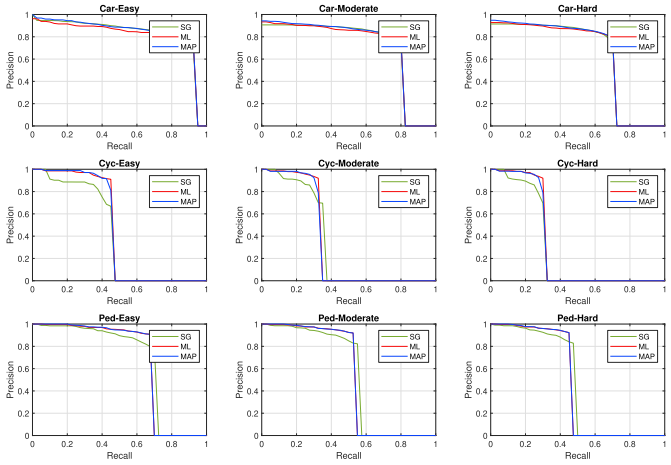


Fig. 13. Precision-recall curves for RaV modality, with $\lambda_{ML} = 1.3 \times 10^{-3}$, $Bins_{ML} = 20$, $\lambda_{MAP} = 1.7 \times 10^{-5}$, and $Bins_{MAP} = 24$.

In fact, the upsample is for estimating points at positions where there are no projected points. The estimate of such

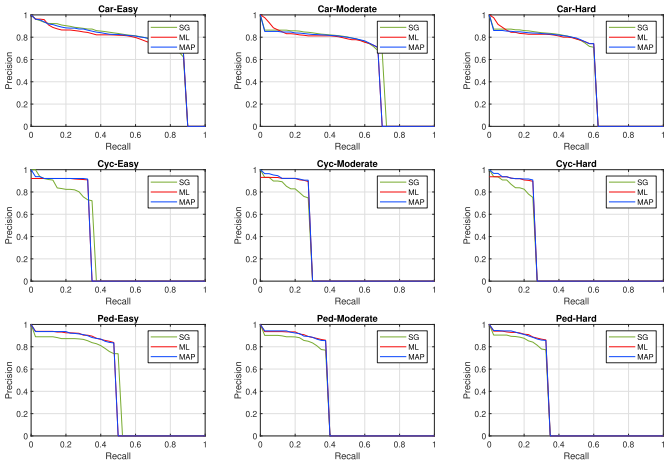


Fig. 14. Precision-recall curves for ReV modality, with $\lambda_{ML} = 1.3 \times 10^{-3}$, $Bins_{ML} = 23$, $\lambda_{MAP} = 8.0 \times 10^{-5}$, and $Bins_{MAP} = 5$.

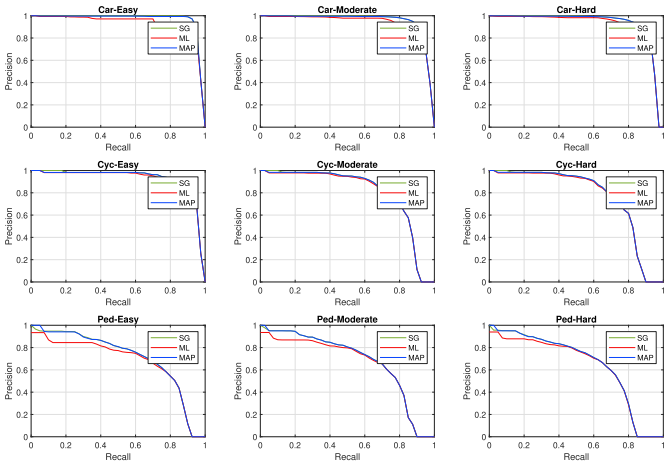


Fig. 15. Precision-recall curves considering $2D$ bounding boxes after SECOND detector training.

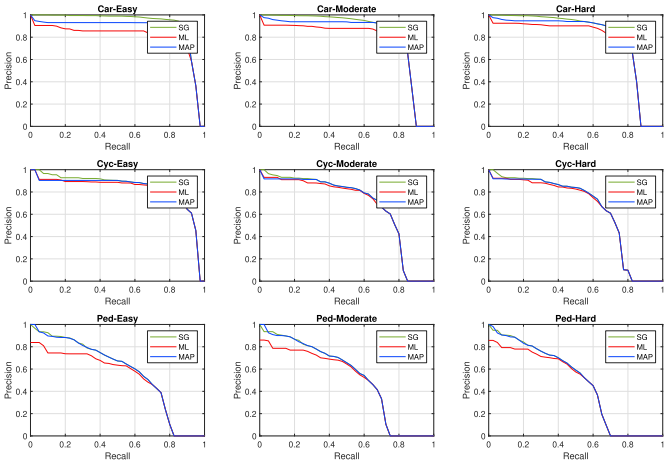


Fig. 16. Precision-recall curves using the SECOND detector to detect $3D$ bounding boxes.

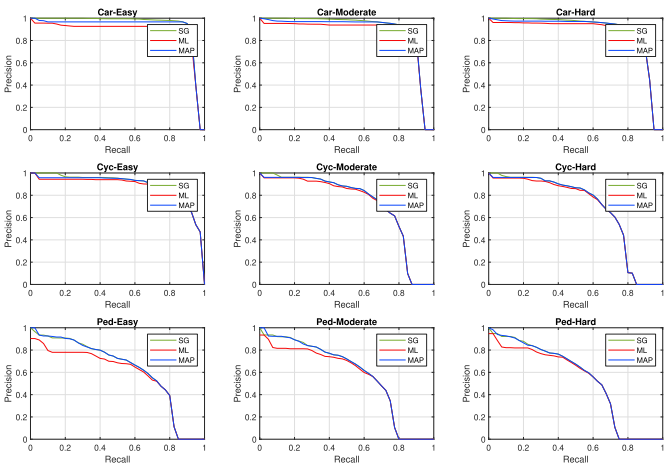


Fig. 17. Precision-recall curves considering BEV detection.

points can be performed by considering a mask C_{mask} of size $c \times c$ pixels, and using the sliding window principle. The sampled point $\hat{r_0}$, located at the center of C_{mask}, is weighted by the number of neighboring points defined by the mask size *i.e.*, the formulation combines the intensity and distance values of a pixels group which are inside the mask C_{mask}, being $c_0 = (c_h, c_v)$ the mask center, which is the localization of interest, and $\hat{r_0}$ the value to be estimated at c_0 from the r_i (RaV or ReV), where c_h and c_v are the positions in the horizontal and vertical directions respectively, as in Fig. 11.

V. Experiments and Results

In this section, we evaluate quantitatively the proposed approach to reduce overconfident predictions through the *ML* and *MAP* layers, considering Gaussian distributions, and normalized histograms, to model the prior and likelihood respectively. The approach depends of some "hyperparameters" that interfere in the results achieved by the ML and MAP layers. The additive smoothing λ (c.f. Sect. III-A), the chosen densities *e.g.*, the numbers of bins of the normalized histograms (described in Sect. III-B above), are design dependent parameters and hence are subjected to the problem in hands. Here, the choice of these parameters has been made experimentally.

The experiments conducted in this Section to assess the proposed technique and to support comparison studies make use of the KITTI 'Object Detection' dataset,[4] both the RGB (camera) and the LiDAR modalities (necessary for the RaV, ReV, and $3D$ point cloud). We have split the original training set by considering 3367 frames for training, 375 for validation, and then the remaining 3739 frames comprise the actual test set. RGB, RaV, and ReV modalities were trained with the same hyperparameters (learning rate, image size, anchors, strides, IoU threshold, etc.) for YOLOV4, while the $3D$ point clouds were trained directly via the SECOND detector.

A. General Performance and Overconfidence

The results on the *per-modalities* test sets are shown in figures 12, 13, and 14 through precision-recall curves (Pr-Rc) for YOLOV4, while the figures 15, 16, and 17 correspond to the experimental results achieved with the SECOND detector.

[4]http://www.cvlibs.net/datasets/kitti/eval_3dobject.php

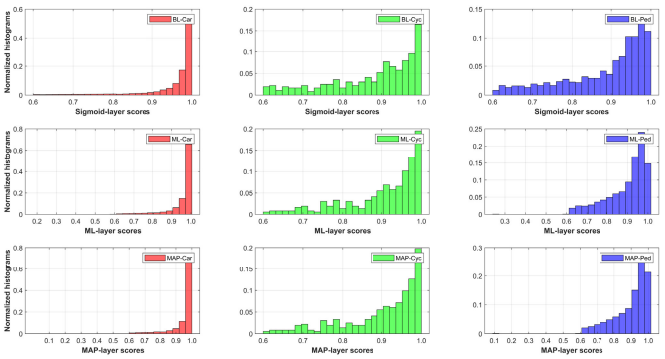
Note that the curves are presented to the three different difficulty levels (easy, moderate and hard), according to the KITTI dataset methodology for object detection.

In addition to the results given by the Pr-Rc curves, we further present a quantitative comparison, between the baseline (designated by Sigmoid, or simply *SG*) and the proposed *ML*, and *MAP* layers, using the area under the curve (AUC), as shown in tables II and IV.

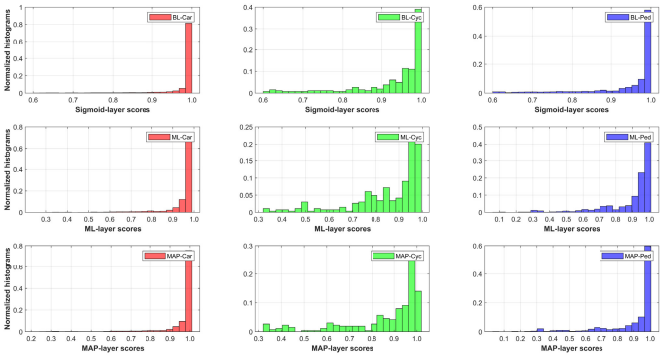
Based on the Pr-Rc curves using YOLOV4, it is possible to observe that the proposed probabilistic inference (*ML*, and *MAP* layers) outperformed the baseline (*SG* layer) in almost all modalities and for most of the difficulty levels, particularly for the cyclist class, which has the smallest amount of objects in both training and test sets. To facilitate the comparison analysis, Table II contains the AUC from these experiments, where the best achieved detection performances are highlighted in bold. The AUC metrics show that ML and MAP achieved very satisfactory performance for different levels of difficulties and classes, as well as for different modalities. Additionally, the graphs in figures 18 and 19 show, when using the YOLOV4 detector, the distribution of the output-scores for the proposed approach and the baseline (*i.e.*, using Sigmoid). We can see that the baseline results achieved by YOLOV4 (shown in the first row) present many false positives (FP) with overconfident scores, while the *ML* and *MAP* layers have reduced the overconfidence on the FPs, whereas the performance on the true positives (TP) is relatively unaffected, according to Table III.

The SECOND detector receives $3D$ point-clouds as input thus, besides $3D$ detection, we have converted the $3D$ representation to $2D$ and Bird's Eye View (BEV) for completeness of the results and benchmarking analysis. At first glance, the *ML*, and *MAP* approaches when applied to SECOND demonstrate to be less effective in improving the detection performance. This is due to the small amount of high-scoring (*i.e.*, highly confident) false positives in SECOND, as can be analyzed in Fig. 20 - this is more evident on the car category. Conversely, a bigger overlap of a relatively less distinguishable score range (0.4-0.6) can be improved by reweighing the scores. In this way, the probabilistic approach proposed in this work was applied to perform a 'smoothing' on the classification scores to mitigate overconfidence, as can be seen from Fig. 20, regarding the pedestrian class. Overall, we can say that the results achieved by the *ML* and *MAP* layers for the car and cyclist categories showed quite similar results compared to the baseline. Such results can be seen in Table IV, this implies that the approach may compromise slightly the overall performance. The ML and MAP layers were compiled considering $\lambda_{ML} = 5 \times 10^{-3}$, $Bins_{ML} = 22$, $\lambda_{MAP} = 1 \times 10^{-4}$, and $Bins_{MAP} = 24$.

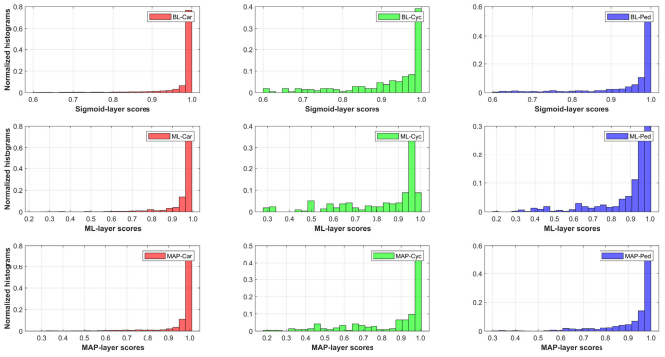
The proposed technique for the SECOND detector tends to perform better on the 'hard' level objects. We can conclude that, because the baseline implementation on SECOND does not attained overconfident behaviour, as shown by the results, the proposed approach degraded a bit the overall performance for that particular detector but, on the other hand, it smoothed the scores for the false positives (which is very desirable in autonomous driving), according to Table V. Furthermore, the proposed approach has the advantage of giving probabilistic interpretation to the detectors.



(a) RGB modality.



(b) RaV modality.



(c) ReV modality.

Fig. 18. Score distributions considering TP objects from YOLOV4 detector.

As the SECOND detector provides a relatively regularized scores across the classes, the *ML* and *MAP* approaches have limited improvement by eliminating the high-scoring FPs. However, the probabilistic approach is able to distinguish the ambiguous scores from the pedestrian class. This can be shown by the more overlap score range of true and false positive objects (Figures 15, 16, 17, and 20).

B. Calibration Error

Typically, the calibration of probabilistic predictions (which relates the model's prediction scores to the true correctness likelihood [79]) is analyzed by the Expected Calibration Error (ECE) metric [31]. The ECE is obtained from a histogram with M bins, where each bin contains a group of scores (predicted

TABLE II

AUC, IN %, FOR THE BASELINE METHOD DENOTED BY *SG*, AND THE PROPOSED APPROACHES (*ML* AND *MAP* LAYERS). THERESULTS REFER TO THE TRUE POSITIVES AND HAVE BEEN ACHIEVED BY THE YOLOV4 IMPLEMENTATION USING 2*D* REPRESENTATIONS

RGB Modality											
	Easy				**Moderate**				**Hard**		
Case	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***
Car	75.48	75.93	**75.95**	**Car**	70.67	70.90	**71.00**	**Car**	63.04	**63.36**	**63.36**
Cyc	45.47	**47.20**	**47.20**	**Cyc**	45.47	46.83	**46.99**	**Cyc**	40.94	42.09	**42.22**
Ped	61.84	**63.05**	**63.05**	**Ped**	**52.27**	51.25	51.24	**Ped**	**45.65**	44.52	44.52
RaV Modality											
	Easy				**Moderate**				**Hard**		
Case	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***
Car	82.99	81.13	**83.21**	**Car**	71.07	72.16	**71.78**	**Car**	62.97	62.80	**63.53**
Cyc	40.48	**44.80**	44.73	**Cyc**	32.28	**32.74**	32.43	**Cyc**	28.13	**30.39**	29.99
Ped	66.27	66.45	**66.60**	**Ped**	**52.56**	52.22	52.22	**Ped**	**45.57**	44.93	44.96
ReV Modality											
	Easy				**Moderate**				**Hard**		
Case	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***
Car	**74.42**	72.68	73.92	**Car**	**58.13**	56.14	56.35	**Car**	50.83	50.69	50.52
Cyc	30.80	31.00	**31.25**	**Cyc**	24.65	26.46	**26.86**	**Cyc**	22.73	24.21	**24.53**
Ped	43.51	**44.35**	44.26	**Ped**	33.62	35.44	**35.45**	**Ped**	29.32	**30.88**	30.87

TABLE III

THE AVERAGE OF THE SCORES WITH THE PROPOSED APPROACH, CONSIDERING THE RESULTS FROM THE YOLOV4

	Modality	**RGB**			**RaV**			**ReV**		
True Positives	Approach	*SG*	*ML*	*MAP*	*SG*	*ML*	*MAP*	*SG*	*ML*	*MAP*
	Average	0.947	0.950	0.950	0.974	0.940	0.955	0.970	0.934	0.951
	Variance	0.007	0.006	0.006	0.004	0.010	0.011	0.005	0.011	0.012
	Modality	**RGB**			**RaV**			**ReV**		
False Positives	Approach	*SG*	*ML*	*MAP*	*SG*	*ML*	*MAP*	*SG*	*ML*	*MAP*
	Average	0.788	0.806	0.806	0.867	0.780	0.786	0.872	0.795	0.817
	Variance	0.013	0.013	0.013	0.015	0.037	0.044	0.014	0.034	0.030

TABLE IV

AUC FOR THE *SG*, *ML* AND *MAP* LAYERS, USING THE SECOND DETECTOR, CONSIDERING THE TRUE-POSITIVE OBJECTS

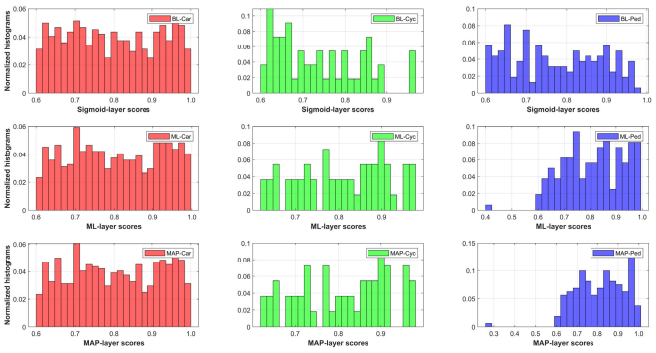
2D Detection											
	Easy				**Moderate**				**Hard**		
Case	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***
Car	**96.88**	93.09	96.57	**Car**	**95.42**	93.61	95.24	**Car**	**93.02**	91.89	92.88
Cyc	**92.66**	91.91	92.44	**Cyc**	**80.27**	79.65	80.14	**Cyc**	**76.65**	76.11	76.52
Ped	70.77	67.22	**70.87**	**Ped**	67.74	65.35	**67.78**	**Ped**	64.09	62.36	**64.16**
3D Detection											
	Easy				**Moderate**				**Hard**		
Case	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***
Car	**91.80**	79.40	87.27	**Car**	**82.86**	75.57	80.32	**Car**	**79.86**	75.16	78.15
Cyc	**84.21**	81.89	82.88	**Cyc**	**67.99**	66.59	67.31	**Cyc**	**64.03**	62.80	63.50
Ped	**57.19**	51.45	57.11	**Ped**	52.39	48.60	**52.41**	**Ped**	**47.42**	44.43	47.38
BEV Detection											
	Easy				**Moderate**				**Hard**		
Case	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***	**Case**	***SG***	***ML***	***MAP***
Car	**93.67**	86.44	91.55	**Car**	**89.81**	85.64	88.49	**Car**	**88.90**	86.24	88.02
Cyc	**89.30**	87.24	88.59	**Cyc**	**72.41**	71.17	72.04	**Cyc**	**68.14**	67.07	67.85
Ped	61.98	57.26	**62.07**	**Ped**	57.82	54.83	**57.89**	**Ped**	53.39	51.11	**53.41**

TABLE V

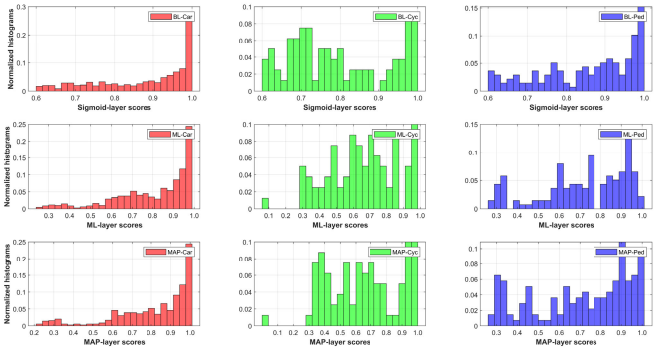
THE AVERAGE OF THE SCORES AFTER THE PROPOSED APPROACH, CONSIDERING THE RESULTS FROM THE SECOND DETECTOR FOR 3*D* POINT CLOUDS

True Positives	Approach	*SG*	*ML*	*MAP*	False Positives	Approach	*SG*	*ML*	*MAP*
	Average	0.860	0.570	0.310		Average	0.258	0.161	0.091
	Variance	0.030	0.017	0.008		Variance	0.026	0.017	0.005

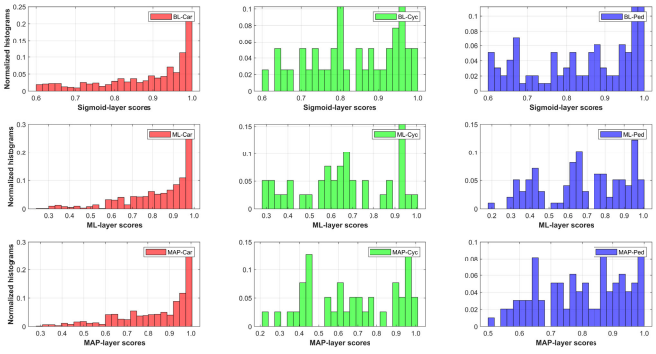
values). Each object with its respective classification score is allocated within a bin, according to the prediction confidence *i.e.*, maximum prediction value. Each bin B_m is defined through a range $I_m = \left(\frac{(m-1)}{M}, \frac{m}{M}\right]$, where $m = 1, .., M$. The average accuracy - $acc(B_m)$ - is obtained for each bin B_m, as well as the average confidence $conf(B_m) = \frac{1}{|B_m|}\sum_i \hat{p}_i$,



(a) RGB modality.

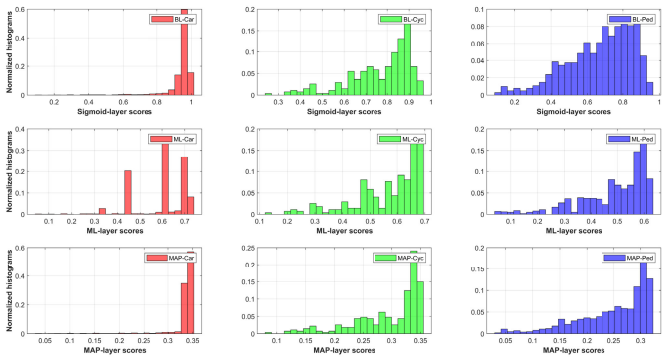


(b) RaV modality.

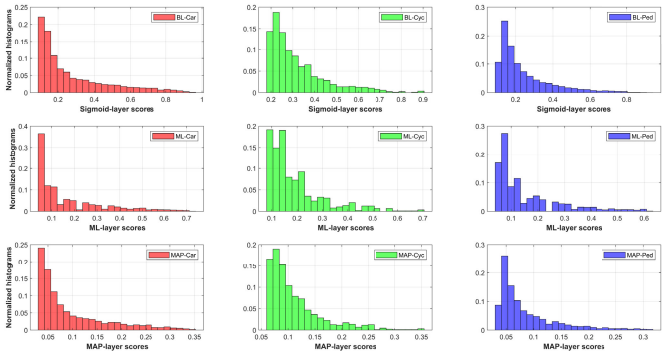


(c) ReV modality.

Fig. 19. Score distributions considering FP objects from YOLOV4 detector.



(a) TP objects.



(b) FP objects.

Fig. 20. Score distributions considering objects from SECOND detector.

where $\hat{p}_i$ is the confidence for classified object i and $|B_m|$ is the amount of objects in each bin B_m. From the $acc(\cdot)$ and $conf(\cdot)$, the ECE is obtained according to (13):

$$ECE = \sum_{m=1}^{M} \frac{|B_m|}{n} |acc(B_m) - conf(B_m)|, \qquad (13)$$

where n is the total the number of objects. Thus, the proposed approach can be compared quantitatively with the baseline through the ECE, as shown in Table VI (RGB, RaV and ReV modalities) and Table VII ($3D$ Point clouds). Based on the results shown in Table VI, considering the YOLOV4 detector, we can see that the ECE was reduced for the proposed methodology. However, for the SECOND detector applied to point-cloud representation the achieved ECE remained close to the baseline - as shown in Table VII.

TABLE VI

ECE ON THE DIFFERENT MODALITIES, WHEN USING YOLOV4 AS DETECTOR

	RGB Modality		
Method:	**SG (baseline)**	**ML**	**MAP**
ECE	0.007	**0.005**	**0.005**
	RaV Modality		
Method:	**SG (baseline)**	**ML**	**MAP**
ECE	0.036	**0.013**	0.027
	ReV Modality		
Method:	**SG (baseline)**	**ML**	**MAP**
ECE	0.031	**0.013**	0.031

TABLE VII

ECE FOR THE DETECTOR SECOND - $3D$ POINT CLOUDS

	3D - PointCloud		
Method:	**SG (baseline)**	**ML**	**MAP**
ECE	**0.196**	0.323	0.208

VI. CONCLUDING REMARKS

Many machine learning models, particularly deep learning ones, have the tendency of regarding the values of the detected objects' scores as being a degree of confidence (or related to a probability) without any level of uncertainty *i.e.*, many deep models are not formulated to provide uncertainties associated with the predicted results. One way to ensure that the classification scores of detected objects can be interpreted as probabilistic values or have some level of uncertainty is through calibration/regularization techniques. However, the developments of such techniques are quite challenging, for

instance because there is no ground truth available on uncertainty data - and it is still an open problem.

The state-of-the-art formalism to capture model uncertainties (calibration/regularization techniques), during training or at the time test phase, aim to ensure confidence measures for the predictions of the models. In this way, this paper proposes a formulation considering the concepts of Maximum Likelihood (ML) and Maximum a-Posteriori (MAP) to reduce the overconfidence of detected false positive objects from the classification scores *i.e.*, the ML/MAP layers are be able to reduce confidence in incorrect predictions. The formulation takes into account a probabilistic inference through two models, one being non-parametric (normalized histogram) and the other is parametric (Gaussian density to model the priors for the MAP).

As a way to present the efficiency of the proposed probabilistic inference approach, this work considered different modalities, as RGB imagens, RaV, and ReV maps, as well as $3D$ point clouds data *i.e.*, datasets with different characteristics. In the case of RGB images, the characteristics are obtained directly from the camera, while RaV and ReV maps are obtained from depth (range-view) and intensity (reflectance-view) data, respectively. In addition, this paper has considered the detection of objects directly on 3D point clouds, as input, processed by a LiDAR-based pipeline - SECOND [77].

The results achieved by the proposed approach are very satisfactory, specially for the cyclists class (for YOLOV4), and pedestrian case (for SECOND), as evidenced by the improvements in general performance (evaluated with the Pr-Rc curves and AUC), reduction of overconfidence (illustrated in Figures 18, 19 and 20) and a general reduction in the calibration error (evaluated using the ECE). Finally, a key advantage of the proposed approach is that there is no need to perform a new network training, that is, the approach has been applied on already trained networks.

REFERENCES

[1] G. Singh et al., "ROAD: The road event awareness dataset for autonomous driving," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 1, pp. 1036–1054, Jan. 2023.

[2] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2D and 3D," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 45, no. 3, pp. 3292–3310, Mar. 2023.

[3] Q. He et al., "Stereo RGB and deeper LiDAR-based network for 3D object detection in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 1, pp. 152–162, Jan. 2023.

[4] J. Janai, F. Güney, A. Behl, and A. Geiger, "Computer vision for autonomous vehicles: Problems, datasets and state of the art," *Found. Trends Comput. Graph. Vis.*, vol. 12, nos. 1–3, pp. 201–308, 2020.

[5] S. Liu, L. Li, J. Tang, S. Wu, and J.-L. Gaudiot, "Creating autonomous vehicle systems," *Synth. Lectures Comput. Sci.*, vol. 6, no. 1, pp. 1–186, 2017.

[6] L. Claussmann, M. Revilloud, D. Gruyer, and S. Glaser, "A review of motion planning for highway autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1826–1848, May 2020.

[7] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 year, 1000km: The Oxford robotcar dataset," *Int. J. Robot. Res.*, vol. 36, no. 1, pp. 3–15, 2017.

[8] S. Aly, "Partially occluded pedestrian classification using histogram of oriented gradients and local weighted linear kernel support vector machine," *IET Comput. Vis.*, vol. 8, no. 6, pp. 620–628, 2014.

[9] D. Su, H. Zhang, H. Chen, J. Yi, P.-Y. Chen, and Y. Gao, "Is robustness the cost of accuracy—A comprehensive study on the robustness of 18 deep image classification models," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 1–19.

[10] A. Bochkovskiy, C. Wang, and H. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*.

[11] E. Zhang and Y. Zhang, *F-Measure*. Boston, MA, USA: Springer, 2009, p. 1147.

[12] C. Goutte and E. Gaussier, "A probabilistic interpretation of precision, recall and f-score, with implication for evaluation," in *Proc. 27th Eur. Conf. Adv. Inf. Retr. Res.* Berlin, Germany: Springer-Verlag, 2005, Paper 345b359.

[13] Y. LeCun et al., "Backpropagation applied to handwritten zip code recognition," *Neural Comput.*, vol. 1, no. 4, pp. 541–551, 1989.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012, pp. 1–12.

[15] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.

[16] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, vol. 97, Jun. 2019, pp. 6105–6114.

[17] A. Dosovitskiy et al., "An image is worth 16×16 words: Transformers for image recognition at scale," in *Proc. 9th Int. Conf. Learn. Represent.*, 2021, pp. 1–15.

[18] I. O. Tolstikhin et al., "MLP-mixer: An all-MLP architecture for vision," 2021, *arXiv:2105.01601*.

[19] R. McAllister et al., "Concrete problems for autonomous vehicle safety: Advantages of Bayesian deep learning," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, 2017, pp. 4745–4753.

[20] D. Feng, A. Harakeh, S. L. Waslander, and K. Dietmayer, "A review and comparative study on probabilistic object detection in autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 9961–9980, Aug. 2021.

[21] G. Melotti, C. Premebida, J. J. Bird, D. R. Faria, and N. GonC'alves, "Probabilistic object classification using CNN ML-MAP layers," in *Proc. Eur. Conf. Comput. Vis. Workshop Perception Auto. Driving*, 2020, pp. 1–7.

[22] D. Feng et al., "Labels are not perfect: Inferring spatial uncertainty in object detection," *IEEE Trans. Intell. Transp. Syst.*, vol. 8, no. 3, pp. 1–14, Dec. 2021.

[23] D. Feng, L. Rosenbaum, F. Timm, and K. Dietmayer, "Labels are not perfect: Improving probabilistic object detection via label uncertainty," in *Proc. Eur. Conf. Comput. Vis. Workshop Perception Auto. Driving*, 2020, pp. 1–5.

[24] R. Patra, R. Hebbalaguppe, T. Dash, G. Shroff, and L. Vig, "Calibrating deep neural networks using explicit regularisation and dynamic data pruning," in *Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2023, pp. 1541–1549.

[25] R. Krishnan and O. Tickoo, "Improving model calibration with accuracy versus uncertainty optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 18237–18248.

[26] D. P. P. Mesquita, L. A. Freitas, J. P. P. Gomes, and C. L. C. Mattos, "LS-SVR as a Bayesian RBF network," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 4389–4393, Oct. 2019.

[27] N. Passalis, M. Tzelepi, and A. Tefas, "Probabilistic knowledge transfer for lightweight deep representation learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 5, pp. 2030–2039, Jun. 2020.

[28] K. Posch and J. Pilz, "Correlated parameters to accurately measure uncertainty in deep neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 3, pp. 1037–1051, Mar. 2021.

[29] D. Feng, L. Rosenbaum, F. Timm, and K. Dietmayer, "Leveraging heteroscedastic aleatoric uncertainties for robust real-time LiDAR 3D object detection," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 1280–1287.

[30] Y. Zou, Z. Yu, X. Liu, B. V. K. V. Kumar, and J. Wang, "Confidence regularized self-training," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5981–5990.

[31] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 1321–1330.

[32] G. Pereyra, G. Tucker, J. Chorowski, Å. Kaiser, and G. Hinton, "Regularizing neural networks by penalizing confident output distributions," 2017, *arXiv:1701.06548*.

[33] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," in *Proc. 33rd Int. Conf. Mach. Learn. (PMLR)*, vol. 48, 2016, pp. 1050–1059.

[34] D. P. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28. Red Hook, NY, USA: Curran Associates, 2015, pp. 1890–1983.

[35] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra, "Weight uncertainty in neural network," in *Proc. ICML*, vol. 37, 2015, pp. 1613–1622.

[36] D. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. 2nd Int. Conf. Learn. Represent. (ICLR)*, 2014, pp. 1–16.

[37] A. Graves, "Practical variational inference for neural networks," in *Proc. 24th Int. Conf. Neural Inf. Process.*, vol. 24, Dec. 2011, pp. 2348–2356.

[38] J. Cheng and N. Vasconcelos, "Calibrating deep neural networks by pairwise constraints," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 13699–13708.

[39] L. Frenkel and J. Goldberger, "Network calibration by temperature scaling based on the predicted confidence," in *Proc. 30th Eur. Signal Process. Conf. (EUSIPCO)*, Aug. 2022, pp. 1586–1590.

[40] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2002, pp. 694–699.

[41] J. C. Platt, "Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods," in *Proc. Adv. Large Margin Classifiers*, 2000, pp. 61–74.

[42] W. Li, G. Dasarathy, and V. Berisha, "Regularization via structural label smoothing," in *Proc. 23rd Int. Conf. Artif. Intell. Statist.*, vol. 108, Aug. 2020, pp. 1453–1463.

[43] R. Müller, S. Kornblith, and G. E. Hinton, "When does label smoothing help?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds. Red Hook, NY, USA: Curran Associates, 2019, pp. 1–190.

[44] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?" in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 5574–5584.

[45] C. M. Bishop, *Pattern Recognition and Machine Learning*. Cham, Switzerland: Springer, 2006.

[46] P. Conde and C. Premebida, "Adaptive-TTA: Accuracy-consistent weighted test time augmentation method for the uncertainty calibration of deep learning classifiers," in *Proc. 33rd Brit. Mach. Vis. Conf.* London, U.K.: BMVA Press, Nov. 2022, pp. 782–787.

[47] A. Kristiadi, M. Hein, and P. Hennig, "Being bayesian, even just a bit, fixes overconfidence in ReLU networks," 2020, *arXiv:2002.10118*.

[48] S. Thulasidasan, G. Chennupati, J. A. Bilmes, T. Bhattacharya, and S. Michalak, "On mixup training: Improved calibration and predictive uncertainty for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 13888–13899.

[49] K. B. Bulatov and D. V. Polevoy, "Reducing overconfidence in neural networks by dynamic variation of recognizer relevance," in *Proc. 29th Eur. Conf. Model. Simul.*, 2015, pp. 488–491.

[50] V. S. Raudys, R. Somorjai, and R. Baumgartner, "Reducing the overconfidence of base classifiers when combining their decisions," in *Multiple Classifier Systems*. Boston, MA, USA: Springer, 2003, pp. 65–73.

[51] 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.

[52] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse, "Flipout: Efficient pseudo-independent weight perturbations on mini-batches," in *Proc. 6th Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 190–198.

[53] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6402–6413.

[54] M. Lukasik, S. Bhojanapalli, A. Menon, and S. Kumar, "Does label smoothing mitigate label noise?" in *Proc. 37th Int. Conf. Mach. Learn.*, vol. 119, 2020, pp. 6448–6458.

[55] Y. Gal, J. Hron, and A. Kendall, "Concrete dropout," in *Proc. 31st Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1–28.

[56] L. Neumann, A. Zisserman, and A. Vedaldi, "Relaxed softmax: Efficient confidence auto-calibration for safe pedestrian detection," in *Proc. NIPS Workshop Machine Learn. Intell. Transp. Syst.*, 2018, pp. 180–187.

[57] C. Corbière, N. Thome, A. Bar-Hen, M. Cord, and P. Pérez, "Addressing failure prediction by learning model confidence," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 32, 2019, pp. 1–90.

[58] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–10.

[59] B. Liu, I. B. Ayed, A. Galdran, and J. Dolz, "The devil is in the margin: Margin-based label smoothing for network calibration," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Dec. 2022, pp. 80–88.

[60] Y. Hur, E. Yang, and S. J. Hwang, "A simple framework for robust out-of-distribution detection," *IEEE Access*, vol. 10, pp. 23086–23097, 2022.

[61] Y. Wang, B. Li, T. Che, K. Zhou, Z. Liu, and D. Li, "Energy-based open-world uncertainty modeling for confidence calibration," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 9302–9311.

[62] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014, *arXiv:1412.6572*.

[63] C. Szegedy et al., "Intriguing properties of neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 24–35.

[64] T. DeVries and G. W. Taylor, "Learning confidence for out-of-distribution detection in neural networks," 2018, *arXiv:1802.04865*.

[65] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *Proc. 6th Int. Conf. Learn. Represent.*, 2018, pp. 10–19.

[66] Y. Gal, "Uncertainty in deep learning," Ph.D. dissertation, Univ. Cambridge, Cambridge, U.K., 2016.

[67] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (Adaptive Computation and Machine Learning). Cambridge, MA, USA: MIT Press, 2016.

[68] D. Molchanov, A. Ashukha, and D. Vetrov, "Variational dropout sparsifies deep neural networks," in *Proc. 34th Int. Conf. Mach. Learn. (PMLR)*, vol. 70, 2017, pp. 2498–2507.

[69] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1929–1958, 2014.

[70] A. Papoulis and U. Pillai, *Probability, Random Variables and Stochastic Processes*, 4th ed. New York, NY, USA: McGraw-Hill, Nov. 2001.

[71] D. Valcarce, J. Parapar, and Á. Barreiro, "Additive smoothing for relevance-based language modelling of recommender systems," in *Proc. 4th Spanish Conf. Inf. Retr.*, Jun. 2016, pp. 210–216.

[72] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," *Comput. Speech Lang.*, vol. 13, no. 4, pp. 359–393, Oct. 1999.

[73] G. J. Lidstone, "Note on the general case of the bayes-laplace formula for inductive or a posteriori probabilities," *Trans. Fac. Actuaries*, vol. 8, Dec. 1920, Art. no. 182b192.

[74] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 7, 2020, pp. 12993–13000.

[75] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 345–355.

[76] Z. Yao, Y. Cao, S. Zheng, G. Huang, and S. Lin, "Cross-iteration batch normalization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 1–15.

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

[78] G. Melotti, C. Premebida, and N. Goncalves, "Multimodal deep-learning for object recognition combining camera and LiDAR data," in *Proc. IEEE Int. Conf. Auto. Robot Syst. Competitions (ICARSC)*, Apr. 2020, pp. 177–182.

[79] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, 2005, pp. 625–632.

Gledson Melotti received the bachelor's degree in electrical engineering from the Federal University of São João del-Rei, Minas Gerais, Brazil, in 2006, the master's degree in electrical engineering from the Federal University of Minas Gerais, Minas Gerais, in 2009. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Coimbra, Portugal. His research interests include statistical learning, deep learning, point clouds, and sensor fusion with applications in autonomous driving.

Weihao Lu received the M.Eng. degree in aeronautical engineering from Imperial College London, U.K., in 2018. He is currently pursuing the Ph.D. degree with the Department of Autonomous Systems and Connectivity, University of Glasgow, Glasgow, U.K. His research interests include autonomous driving, 3D point cloud processing, and 3D object detection.

Pedro Conde received the bachelor's degree in mathematics from the NOVA School of Science and Technology, Portugal, and the master's degree in mathematics from the University of Coimbra, Portugal, where he is currently pursuing the Ph.D. degree in electrical and computer engineering (automation and robotics). He is also a Researcher with the Institute of Systems and Robotics, Coimbra, Portugal. His research interests include reliability, uncertainty calibration, and probabilistic interpretation of deep neural networks, with emphasis in applications related to computer vision and autonomous systems.

Dezong Zhao (Senior Member, IEEE) received the B.Eng. and M.S. degrees in control engineering from Shandong University in 2003 and 2006, respectively, and the Ph.D. degree in control engineering from Tsinghua University in 2010. He was a Lecturer of intelligent systems with Loughborough University. Since 2020, he has been a Senior Lecturer of autonomous systems with the University of Glasgow. His research interests include connected and automated vehicles, robotics, machine learning, and control engineering. He has been an EPSRC Innovation Fellow since 2018 and a Royal Society-Newton Advanced Fellow since 2020.

Alireza Asvadi received the Ph.D. degree in electrical and computer engineering from the University of Coimbra, Portugal, in 2018. With over a decade of experience in computer vision, machine learning, and robotics, his primary focus is deploying cutting-edge perception algorithms in the field of robotics. He joined IADYS, France, in 2022, where he is currently a Robot Vision Engineer.

Nuno Gonçalves (Member, IEEE) received the Ph.D. degree in computer vision from the University of Coimbra, Portugal, in 2008. Since 2008, he has been a Tenured Assistant Professor with the Department of Electrical and Computers Engineering, Faculty of Sciences and Technology, University of Coimbra. He is currently a Senior Researcher with the Institute of Systems and Robotics, University of Coimbra. He has been recently coordinating several projects centered on the technology transfer to the industry. In 2018, he joined the Portuguese Mint and Official Printing Office (INCM), where he coordinates innovation projects in areas, such as biometrics, facial recognition, morphing attack detection, graphical security, security coding, and robotics. He has been working in the design and introduction of new products as result of the innovation projects. He is the author of several papers and communications in high-impact journals and international conferences. His research interests include computer vision, visual information security, biometrics, computer graphics, autonomous driving, and robotics.

Cristiano Premebida (Member, IEEE) is currently an Assistant Professor with the Department of Electrical and Computer Engineering (DEEC), University of Coimbra (UC), Portugal, where he is a member of the Institute of Systems and Robotics (ISR-UC). His research interests include autonomous systems, intelligent vehicles, robotic perception, machine learning, sensor fusion, multimodal and multisensory perception for robotics and autonomous systems applications, developing calibration strategies, and probability-prediction approaches to increase robustness of deep models. He is a member of the IEEE ITS Society and the RAS Society. He has been serving as an AE for the various IEEE conferences, such as ITSC, IV, and IROS.