BERGISCHE
UNIVERSITÄT
WUPPERTAL

HOCHSCHULE RUHR WEST
UNIVERSITY OF APPLIED SCIENCES

Uncertainty Calibration and its Application to Object Detection

DISSERTATION

zur Erlangung des Grades
Doctor rerum naturalium

im Fach **Informatik**
an der **Fakultät für Mathematik und Naturwissenschaften**
der **Bergischen Universität Wuppertal**

von
Fabian Thomas Küppers

Erstprüfer	Prof. Dr. rer. nat. Hanno Gottschalk Bergische Universität Wuppertal
Zweitprüfer	Prof. Dr.-Ing. Anselm Haselhoff Hochschule Ruhr West

Mülheim an der Ruhr, Januar 2023

Affidavit

I hereby declare that I have
written this thesis by myself,
only using the sources and
materials as indicated.

Mülheim an der Ruhr, 24. January 2023

Abstract

Image-based environment perception is an important component especially for driver assistance systems or autonomous driving. In this scope, modern neuronal networks are used to identify multiple objects as well as the according position and size information within a single frame. The performance of such an object detection model is important for the overall performance of the whole system. However, a detection model might also predict these objects under a certain degree of uncertainty. In this context, we distinguish between epistemic model uncertainty (e.g., due to a lack of knowledge) and aleatoric data uncertainty (e.g., due to random effects in the input data). These uncertainties might have a large impact on the safety of the whole system.

In this work, we examine the semantic uncertainty (which object type?) as well as the spatial uncertainty (where is the object and how large is it?). We evaluate if the predicted uncertainties of an object detection model match with the observed error that is achieved on real-world data. In the first part of this work, we introduce the definition for confidence calibration of the semantic uncertainty in the context of object detection, instance segmentation, and semantic segmentation. We integrate additional position information in our examinations to evaluate the effect of the object's position on the semantic calibration properties. Besides measuring calibration, it is also possible to perform a post-hoc recalibration of semantic uncertainty that might have turned out to be miscalibrated. Thus, we derive new methods to perform a position-dependent recalibration of possibly uncalibrated uncertainty information. Based on these methods, we introduce the concept of Bayesian confidence calibration which allows to determine the epistemic uncertainty that is inherent in the recalibration methods itself. In this way, it is possible to add a new kind of uncertainty to the overall perception chain.

The second part of this work deals with the spatial uncertainty obtained by a probabilistic detection model. In the scope of regression uncertainty, several definitions for the term of calibration exists which we review and relate to each other within this work. Similar to semantic confidence calibration, it is possible to apply a post-hoc calibration of the spatial uncertainty. We review and extend common calibration methods so that it is possible to obtain parametric uncertainty distributions for the position information in a more flexible way.

In the last part, we demonstrate a possible use-case for our derived calibration methods in the context of object tracking. In contrast to object detection, an object tracker seeks to identify the same objects over a sequence of images over time. We integrate our previously proposed calibration techniques and demonstrate the usefulness of semantic and spatial uncertainty calibration in a subsequent process. We can show that uncertainty calibration leads to a significantly improved object tracking.

In conclusion, in this work we show that common object detection models tend to be miscalibrated. This holds for the semantic as well as for the spatial uncertainty. Our new calibration methods are useful techniques to correct miscalibrated uncertainty estimates and have shown to be a valuable contribution to the overall environment perception process.

Preface & Acknowledgments

The use of the 1st person plural has been the familiar style for writing my previous scientific publications and is also used throughout this thesis, even though the majority of the work has been carried out by myself. This work was conducted in the context of the research project “KI-Absicherung - Safe AI for Automated Driving” at the Ruhr West University of Applied Sciences, Bottrop, Germany in cooperation with Elektronische Fahrwerksysteme GmbH, Gaimersheim, Germany, and Visteon Electronics Germany GmbH, Kerpen, Germany.

First and foremost, I would like to gratefully thank Prof. Dr.-Ing. Anselm Haselhoff from the Ruhr West University of Applied Sciences for giving me the opportunity to work at the Ruhr West University within the “KI-Absicherung” project, for his extensive support, and the fruitful discussions during the last three years. Furthermore, I would like to thank our partner Elektronische Fahrwerksysteme GmbH and especially Jonas Schneider for the excellent collaboration. We started the project “KI-Absicherung” in cooperation with Visteon Elektronics under the leadership of Dr. Andreas Wedel, whom I would also like to thank. Finally, I would like to thank Prof. Dr. Hanno Gottschalk from the University of Wuppertal for his support and for the opportunity to graduate at the University of Wuppertal.

Ich möchte mich bei meinen Freunden, meiner Familie und Kollegen für die großartige Unterstützung in meiner Promotionszeit bedanken. Die Gespräche und die mir entgegengebrachte Unterstützung hat mir sehr geholfen, auch schwierigere Phasen in der Promotionszeit zu überstehen. Ein großes Dankeschön geht an Helena, die sich bereiterklärt hat, mein “yellow-from-the-egg”-Englisch zu korrigieren.

Mein langjähriges Ziel, die Promotion abzuschließen, erfordert ein hohes Maß an Durchhaltevermögen. Ich möchte daher ausdrücklich Nathalie dafür danken, dass sie dieses Durchhaltevermögen mit mir hat. Ohne deine Unterstützung wäre ich nicht so weit gekommen.

Publication Overview

1. Küppers, et al.: "Multivariate Confidence Calibration for Object Detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020 [1].
2. Schwaiger, et al.: "From Black-box to White-box: Examining Confidence Calibration under different Conditions," in *Proceedings of the Workshop on Artificial Intelligence Safety 2021 (SafeAI 2021) co-located with the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021)*, 2021 [2].
3. Küppers, et al.: "Bayesian Confidence Calibration for Epistemic Uncertainty Modelling," in *Proceedings of the 2021 IEEE Intelligent Vehicles Symposium (IV)*, 2021 [3].
4. Küppers, et al.: "Confidence Calibration for Object Detection and Segmentation," in: Fingscheidt, Houben, and Gottschalk (eds.): *Deep Neural Networks and Data for Automated Driving - Robustness, Uncertainty Quantification, and Insights Towards Safety*, Springer Nature Switzerland, pp. 225-250, 2022 [4].
5. Küppers, et al.: "Calibration of Neural Networks for Detection Models (German original: Kalibrierung von Neuronalen Netzen für Detektionsmodelle)," in: VDI Wissensforum GmbH (eds.): *Fahrerassistenzsysteme und automatisiertes Fahren*, VDI Verlag, pp. 49-69, 2022 [5].
6. Küppers, et al.: "Parametric and Multivariate Uncertainty Calibration for Regression and Object Detection," in *European Conference on Computer Vision Workshops*, Springer, 2022, *in press* [6].

Contents

List of Figures

List of Tables

List of Acronyms

PDF	Probability Density Function
PMF	Probability Mass Function
CDF	Cumulative Density Function
ECDF	Empirical Cumulative Density Function
HPDI	Highest Posterior Density Interval
HPDR	Highest Posterior Density Region
ERM	Empirical Risk Minimization
MLE	Maximum Likelihood Estimation
MAP	Maximum a Posteriori
SGD	Stochastic Gradient Descent
SVI	Stochastic Variational Inference
ELBO	Evidence Lower Bound
MCMC	Markov-Chain Monte-Carlo
IoU	Intersection over Union
mIoU	Mean Intersection over Union
mAP	Mean Average Precision
AUPRC	Area under Precision-Recall Curve
ReLU	Rectified Linear Unit
CNN	Convolutional Neural Network
RPN	Region Proposal Network
FPN	Feature Pyramid Network
ECE	Expected Calibration Error
MCE	Maximum Calibration Error
D-ECE	Detection Expected Calibration Error
MMCE	Maximum Mean Calibration Error
NLL	Negative Log Likelihood
MSE	Mean Squared Error

M-QCE	Marginal Quantile Calibration Error
C-QCE	Conditional Quantile Calibration Error
UCE	Uncertainty Calibration Error
ENCE	Expected Normalized Calibration Error
PICP	Prediction Interval Coverage Probability
MPIW	Mean Prediction Interval Width
SGV	Standardized Generalized Variance
NEES	Normalized Estimation Error Squared
NIS	Normalized Innovations Squared
MOT	Multiple Object Tracking
GP	Gaussian Process
RBF	Radial Basis Function
BBQ	Bayesian Binning into Quantiles
ENIR	Ensemble of Near Isotonic Regression

Nomenclature

Statistical Symbols

X, Y	Random variables
$\mathbf{X}, \mathbf{Y}$	Multivariate random variables $\mathbf{X} = (X_1, \ldots, X_K)^\top$ and $\mathbf{Y} = (Y_1, \ldots, Y_K)^\top$ of size K
$\mathbb{P}(X = x)$	Probability for X being x
$\mathbb{E}[X]$	Expectation of X
$\mathrm{Var}[X]$	Variance of X
$\mathrm{Cov}[X, Y]$	Covariance of X and Y
μ_X	Mean of X
σ_X^2, σ_X	Variance and standard deviation of X
$\mathbf{\Sigma}_{X,Y}$	Covariance matrix of X and Y
$f_X(x)$	Probability density function (PDF) of X
$F_X(x)$	Cumulative density function (CDF) of X
$F_X^{-1}(\tau)$	Percent point function (inverse of CDF) of X for any quantile $\tau \in (0, 1)$

Functions

$\Gamma(\alpha)$	Gamma function: $\Gamma(\alpha) = (\alpha - 1)!$ for $\alpha \in \mathbb{R}_{>0}$
$\mathrm{B}(\alpha, \beta)$	Beta function: $\mathrm{B}(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ for $\alpha, \beta \in \mathbb{R}_{>0}$
$\mathrm{B}(\boldsymbol{\alpha})$	Multivariate beta function: $\mathrm{B}(\boldsymbol{\alpha}) = \frac{\prod_{k=1}^{K} \Gamma(\alpha_k)}{\Gamma(\sum_{k=1}^{K} \alpha_k)}$ for $\boldsymbol{\alpha} \in \mathbb{R}_{>0}^K$ with K dimensions
$\delta(x)$	Dirac delta function: $\delta(x) = +\infty$ if $x = 0$ else $\delta(x) = 0$
$\Phi(x)$	Sigmoid function: $\Phi(x) = \frac{\exp(x)}{1+\exp(x)} = \frac{1}{1+\exp(-x)}$ with $x \in \mathbb{R}$
$\Phi_{\mathrm{SM}}(x_k)$	Softmax function: $\Phi_{\mathrm{SM}}(x_k) = \frac{\exp(x_k)}{\sum_{k^*=1}^{K} \exp(x_{k^*})}$ with $x_k \in \mathbb{R}$ and K dimensions
$\mathbb{1}(\cdot)$	Indicator function. Returns 1 if argument is true, else 0
$\mathrm{diag}(x_1, \ldots, x_K)$	Diagonal matrix operator with diagonal elements $x_1, \ldots, x_K$.
$\det(\mathbf{\Sigma})$	Determinant of matrix $\mathbf{\Sigma}$

Probability Distributions

$\text{Bern}(x;\theta)$ Bernoulli distribution: $\text{Bern}(x;\theta) = \theta^x(1-\theta)^{1-x}$ with $x \in \{0,1\}$ and $\theta \in [0,1]$

$\text{Cat}(x;\boldsymbol{\theta})$ Categorical distribution: $\text{Cat}(x;\boldsymbol{\theta}) = \prod_{k=1}^{K} \theta_k^{\mathbb{1}(x=k)}$ with $x \in \{1,\dots,K\}$ and $\theta_k \in [0,1]$

$\mathcal{N}(x;\mu,\sigma^2)$ Univariate normal distribution: $\mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-1}{2\sigma^2}(x-\mu)^2\right)$ where $\sigma^2 \in \mathbb{R}_{>0}$

$\mathcal{N}(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma})$ Mv. normal distribution: $\mathcal{N}(\mathbf{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^K \det(\boldsymbol{\Sigma})}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$

$\text{Beta}(x;\alpha,\beta)$ Beta distribution: $\text{Beta}(x;\alpha,\beta) = \frac{1}{\text{B}(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1}$ where $x \in [0,1]$ and $\alpha,\beta \in \mathbb{R}_{>0}$

$\text{Dir}(\mathbf{x};\boldsymbol{\alpha})$ Dirichlet distribution: $\text{Dir}(\mathbf{x};\boldsymbol{\alpha}) = \frac{1}{\text{B}(\boldsymbol{\alpha})} \prod_{k=1}^{K} x_k^{\alpha_k - 1}$ for $\boldsymbol{\alpha} \in \mathbb{R}_{>0}^K$ with K dimensions

$\text{Cauchy}(x;\vartheta,\gamma)$ Cauchy distribution: $\text{Cauchy}(x;\vartheta,\gamma) = \frac{1}{\pi}\frac{\gamma}{\gamma^2+(x-\vartheta)^2}$ where $\gamma \in \mathbb{R}_{>0}$

Notation for used Random Variables and their Realizations

$\mathbf{X}, \mathbf{x} \in \mathcal{X}$ Input image in space $\mathcal{X}$

$Y, y \in \mathcal{Y}$ Label of an object in $\mathcal{Y} = \{1,\dots,K\}$ with K classes

$\hat{Y}, \hat{y} \in \mathcal{Y}$ Predicted label of an object in $\mathcal{Y}$ obtained by an object detector

$\hat{Y}_j^*, \hat{y}_j^* \in \mathcal{Y}$ Predicted label for each pixel $j \in \mathcal{J}$ obtained by an instance/semantic segmentation model

$\bar{Y}, \bar{y} \in \mathcal{Y}$ Ground-truth label of an object in $\mathcal{Y}$

$\bar{Y}_j^*, \bar{y}_j^* \in \mathcal{Y}$ Ground-truth label for each pixel $j \in \mathcal{J}$

$P, p \in [0,1]$ Semantic confidence information of an object

$\hat{P}, \hat{p} \in [0,1]$ Predicted confidence information of an object obtained by an object detector

$\hat{P}_j^*, \hat{p}_j^* \in [0,1]$ Predicted confidence information for each pixel $j \in \mathcal{J}$ obtained by an instance/semantic segmentation model

$\hat{Q}, \hat{q} \in [0,1]$ Calibrated confidence information of an object obtained by a calibration function

$\mathbf{R}, \mathbf{r} \in \mathcal{R}$ Position information of an object in $\mathcal{R}$, commonly with $\mathbf{R} = (c_x, c_y, w, h)^\top$ where c_x, c_y are the center x and y position and w, h are the width and height, respectively, so that $\mathcal{R} = \mathbb{R}^L$ with L as the size of the box encoding

$\hat{\mathbf{R}}, \hat{\mathbf{r}} \in \mathcal{R}$ Predicted position information of an object in $\mathcal{R}$ obtained by an object detector

$\mathbf{R}_j^*, \mathbf{r} \in \mathcal{R}^*$ Position information for each pixel $j \in \mathcal{J}$ where $\mathcal{R}^* = \mathbb{R}^{L^*}$ with L^* as the size of the position encoding for each pixel

$\bar{\mathbf{R}}, \bar{\mathbf{r}} \in \mathcal{R}$ Ground-truth position information of an object in $\mathcal{R}$

$\widetilde{\mathbf{R}}, \widetilde{\mathbf{r}} \in \mathcal{R}$ Predicted position information of an object in $\mathcal{R}$ obtained by an object tracker

$Z, z \in \mathbb{R}$ Logit of an object detector (output before sigmoid/softmax)

$\hat{M}, \hat{m} \in \{0, 1\}$ Indicator variable that a predicted object matches a ground-truth object

$\hat{\mathbf{S}}, \hat{\mathbf{s}} \in \mathcal{S}$ Aggregated detector output in $\mathcal{S}$ with $\hat{\mathbf{S}} = (\hat{P}, \hat{Y}, \hat{\mathbf{R}})^\top$

Tracking-specific Symbols

$\mathbf{H} \in \mathbb{R}^{L \times L^*}$ Observation matrix to translate from tracker state space to detection space during Kalman filtering, with detector box size L and tracker state size L^*

$\mathbf{F} \in \mathbb{R}^{L^* \times L^*}$ State transition matrix to predict the proceeding state during Kalman filtering

$\boldsymbol{\varrho} \sim \mathcal{N}(0, \boldsymbol{\Lambda})$ Static observation/measurement noise with zero mean and covariance matrix $\boldsymbol{\Lambda} \in \mathbb{R}^{L \times L}$

$\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \boldsymbol{\Psi})$ Static system noise with zero mean and covariance matrix $\boldsymbol{\Psi} \in \mathbb{R}^{L^* \times L^*}$

Miscellaneous Symbols

$\mathcal{D}$ Data set with N samples

$h(\cdot)$ Calibration function for semantic confidence or spatial uncertainty calibration

B_i Single bin used within a binning scheme for Histogram Binning or ECE calculation with $i \in \{1, \dots, I\}$ where I is the total number of bins

$w \in \mathbb{R}_{>0}$ Scale weight used for calibration methods such as Logistic Calibration, Variance Scaling, or GP-Normal

$\mathbf{w} \in \mathbb{R}^K_{>0}$ Scale weight vector used for multivariate calibration methods such as position-dependent Logistic Calibration or multivariate GP-Normal with K dimensions

$\delta \in \mathbb{R}$ Scale bias used for calibration methods such as Logistic Calibration or Beta Calibration

$\mathrm{gp}\big(0, k(\cdot, \cdot), \mathbf{B}\big)$ Gaussian Process with zero mean, kernel function $k(\cdot, \cdot)$ and coregionalization matrix $\mathbf{B}$

1 Introduction

Machine learning models and especially neural networks are used in many applications nowadays, e.g., for image classification [7] or object detection [8, 9]. Besides the advantages of using machine learning models, there are also some challenges that users and developers have to face. In particular, the explainability [10] as well as the reliability [11, 12, 13] are relevant issues especially in the context of safety-critical applications such as autonomous driving [14, 15, 16] or medical diagnosis [17, 18, 19]. In this scope, it is mandatory to be able to comprehend the decision-making process of a neural network especially within self-monitoring procedures to be able to identify critical situations or even induce appropriate fallback solutions. If requested, a neural network will always return a decision on each input, regardless of its own uncertainty. Especially in the case of safety-relevant decisions, it is necessary that the model can output a reliable self-assessment of its own uncertainty so that, in case of doubt, an adequate fallback solution can be applied, e.g., the decision-making can be handed over to a human. In the scope of driver assistance systems and autonomous driving, environment perception is an important part to interact with the surrounding area. Such a vehicle commonly uses multiple kinds of sensors such as cameras [20], radar [21], or LiDAR [22] to construct an environment model. This process is schematically shown in Fig. 1.1. In this thesis, we focus on the environment perception using camera-based object detection algorithms. The process of object detection has rapidly developed in the last years. Classical approaches such as a sliding window object detection using hand-crafted features and filters [23] have been replaced by more efficient deep learning based detection architectures such as Faster R-CNN [8] or Single-Shot detectors [24]. The extended single-shot architectures such as YOLO [25] or RetinaNet [9] provide further improvements in performance and speed. These architectures are based on

Figure 1.1: Environment perception of an intelligent vehicle using different sensors and AI-based perception systems (simluated example) [1]. In this work, we focus on the camera-based environment perception with focus on the object detection task.

deep convolutinal neural networks [26, 27] which are neural networks with many intermediate convolutional operations. A convolutional layer is nothing else but a filter mask that is applied to the preceding input to recognize several features, e.g., edges or textures. The advantage of modern deep learning algorithms is that they do not require a hand-crafted design of the convolutional operations. Instead, the models are able to autonomously learn the relevant aspects of the intermediate filters to detect objects within an image.

First used for image classification [26, 28], deep learning architectures have been transferred to the more complex task of object detection [29, 8, 24, 25, 9]. The task of object detection is a joint task of classification (which type is the object?) and regression (where is the object located? which shape has the object?). The initially proposed R-CNN architecture [29] utilizes a selective search algorithm to generate candidate boxes for possible objects in conjunction with a convolutional neural network to classify and refine these boxes. However, the selective search still leads to a large amount of candidate boxes which makes the network training as well as the inference computational expensive. The advanced architectures Fast R-CNN [30] and Faster R-CNN [8] mitigate this problem by using a more flexible region proposal network to select candidate boxes. In the object proposal stage, fixed image locations are used as prior information for possible object candidates and locations. Afterwards, a region proposal network applies a first selection and refinement of candidate boxes. These candidate boxes are finally used in the second stage of the object detector which applies a classification of the object category as well as a refinement of the object size and position using a separate neural network. Similarly, the Single-Shot detector [24], YOLO [25], and RetinaNet [9] also use fixed image locations as prior information for object candidates and locations. However, these architectures do not use a region proposal network but directly apply the selection, classification, and refinement step to the prior object candidates using a dedicated neural network. These architectures use deep convolutional neural networks to extract the relevant image features. This allows for a refinement and classification of the prior object locations to finally generate the object predictions. Thus, these architectures are known as one-stage detection algorithms and have a very low computational runtime.

A related task to object detection is instance segmentation where the shape of individual objects is of interest in addition to their position and size. With the success of modern object detection architectures, it has also been possible to develop instance segmentation architectures. For example, the Mask R-CNN architecture [31] extends the Faster R-CNN [8] detection model by an additional output that infers the object shape by classifying each pixel within the detected object boxes. In contrast, with semantic segmentation, the association of a pixel with a particular class or segment is determined, but individual objects are not identified in this process. Semantic segmentation architectures such as DeepLabv2 [32] or DeepLabv3+ [33] utilize a deep convolutional neural network to extract relevant features within an input image. The output of these segmentation models is a joint classification of all pixels in the input image that serves as an estimation of the relevant image segments.

Recent works have shown the superior performance of deep learning based object detectors and segmenta-

[1] Image from: Ruhr West University of Applied Sciences and University of Wuppertal: *Automated Mobility: Overview of the technological fundamentals* (German original: *Automatisierte Mobilität: Überblick über die technologischen Grundlagen*), p. 13, Fig. 5. © 2022 camo.nrw. Reprinted, with permission.

tion models over the classical approaches [8, 24, 31, 9, 33]. However, driver assistance systems and especially autonomous driving are highly safety-relevant areas in which these detection algorithms need to operate. As already stated, the explainability and the reliability of deep learning based methods are still open fields of current research [11, 12, 13, 10]. A detection model needs to indicate in any case, if it is uncertain about a certain prediction. For example, if a detection model needs to infer multiple objects within a frame under challenging conditions (e.g., rain or with sensors of low quality), the estimation either of the semantic class (what kind of object?) or of the position and size might be difficult and are subject to a high uncertainty.

Modern deep-learning based detection and segmentation architectures are able to estimate their prediction uncertainty during inference. This uncertainty can be interpreted as a statistical measure for the estimation of the prediction error. However, recent works have shown that modern neural networks tend to be too self-confident in their predictions [11, 12, 13, 34], i.e., the estimated prediction uncertainty is too low compared with the observed error. However, obtaining reliable uncertainty information is crucial especially for environment perception within the safety-relevant context of autonomous driving.

In the past, several works address the problem of unreliable uncertainty information by applying an additional calibration step during inference [35, 36, 13, 34]. For classification, the authors in [35] initially sought for a probabilistic output of support vector machines by utilizing the sigmoid function to transform an unrestricted output score to a score in the $[0, 1]$ interval that can be interpreted as an estimated probability of correctness [35]. In common literature, this method is known as logistic calibration or Platt scaling and can be used as a recalibration function to obtain calibrated probabilities [35, 13]. As opposed to this approach, the authors in [36] proposed a binning scheme to convert uncalibrated probabilities to calibrated ones conditioned on the estimated uncertainty. This approach is known as histogram binning. An extension to this approach is Bayesian Binning into Quantiles (BBQ) proposed by [12] which utilizes multiple weighted histogram binning schemes to obtain calibrated probabilities. Furthermore, the authors in [12] proposed the Expected Calibration Error (ECE) and Maximum Calibration Error (MCE) which are both metrics to quantify the misalignment between estimated and observed error. Based on this research, the authors in [13] found that modern network architectures in particular tend to be too self-confident in their predictions, leading to a high calibration error. For multiclass classification tasks, the authors in [13] proposed temperature scaling which is related to the logistic calibration function [35] but only with a single rescaling parameter for all possible output classes. Recently, the authors in [37] and [38] investigated the calibration properties of modern segmentation and object detection architectures, respectively. For both tasks, the authors studied the influence of miscellaneous factors (e.g., position and shape in object detection) to the calibration properties.

In contrast to classification, the task of regression is to estimate a continuous output score, e.g., the object location or shape when used within object detection. In probabilistic regression, a model does not only estimate the requested regression score but also an additional uncertainty that indicates the belief of the estimator in the correctness of its prediction [39, 40]. Recent works found that these uncertainty scores also tend to miscalibration [34, 41, 42, 18] and proposed several methods such as Isotonic Regression [34], Variance Scaling [42, 18], and GP-Beta [41] that are applied after inference as an additional calibration step. A detailled explanation of the calibration methods for probabilistic regression is given in Chap. 5.

1.1 Research Question and Novelty

In this thesis, we address the problem of reliable uncertainty information obtained by modern detection algorithms. We seek to examine if the estimated uncertainty is trustworthy, i.e., if it matches the observed error in real-world applications. We follow this research question and apply our examinations to the semantic (label) uncertainty as well as to the spatial (position and size) uncertainty. Moreover, recent works have shown that modern neural networks tend to produce unreliable uncertainty information [11, 12, 43, 13]. Since the deviation between estimated uncertainty and observed error undergoes a systemic error, it is possible to apply post-hoc calibration methods [12, 13]. Such calibration methods apply a remapping of the estimated uncertainties to more realistic ones that have been observed on a dedicated calibration data set. We transfer these methods to the task of object detection and address the research question, if correlations exist between semantic (label) and spatial (position and size) uncertainty. Thus, we derive new calibration methods that are able to capture possible correlations by using further influential factors for uncertainty calibration within the scope object detection. Our newly proposed methods are evaluated on state-of-the-art detection architectures Faster R-CNN [31] and RetinaNet [9] and the data sets MS COCO [44] and Cityscapes [45].

For environment perception in the scope of driver assistance systems or autonomous driving, the detection of single objects is the first step to implement an object tracking over subsequent frames within a sequence of images. The object detection is thus the baseline task for object tracking [46, 47]. Especially for object tracking, reliable uncertainties are crucial, as they are processed by the tracking algorithm and thus have a significant impact on the tracking performance of individual objects. Therefore, we investigate if our developed calibration methods have an influence on the task of object tracking. In this context, we apply our new uncertainty calibration on the output of the baseline object detector and pass the recalibrated uncertainty information to the tracking algorithm. This process is schematically shown in Fig. 1.2. In this way, we are able to evaluate the influence of the predictive uncertainty in conjunction with our proposed calibration methods on a subsequent process. Our contributions within this work are summarized in Tab. 1.1.

1.2 Structure of this Work

This work is structured as follows: First, we introduce the basic architectures for object detection and how uncertainty information can be extracted from existing object detection architectures in Chap. 2. In Chap. 3, we review the definitions for semantic confidence calibration and transfer these to the task of object detection, instance segmentation, and semantic segmentation. We develop metrics to evaluate the uncertainty as well as methods to correct unreliable confidence information. Furthermore, we evaluate our proposed metrics and methods using several detection and segmentation models on common public data sets. Building on top of this, we propose Bayesian confidence calibration in Chap. 4 that allows to capture additional uncertainty within the uncertainty calibration methods itself. This adds an additional layer to model uncertainty in the overall perception chain and allows a detection of possible failure modes during inference. In Chap. 5, we review the definitions for spatial uncertainty evaluation. In addition, we extend common metrics as well

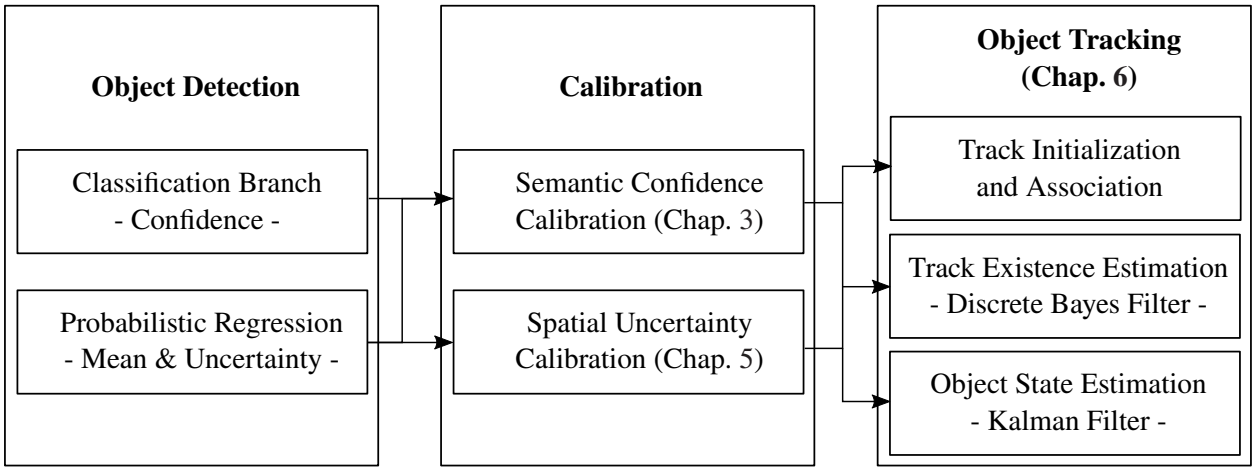


Figure 1.2: Concept of the environment perception pipeline that is addressed within this work. An object detection model generates multiple predictions for possible object candidates with a certain semantic and spatial uncertainty. We examine the semantic and spatial uncertainty for reliability. Furthermore, we provide recalibration methods that can be applied and integrated into the tracking pipeline after the detection process. The recalibrated uncertainty information is finally consumed by a tracking framework to track the detected objects over time in a sequence of images.

Table 1.1: Overview of our contributions within this work.

Contribution (short description)	Reference	Section
Definition of semantic confidence calibration for object detection	[1, 4, 2]	3.2.2
Definition of semantic confidence calibration for instance segmentation	[4]	3.2.3
Definition of semantic confidence calibration for semantic segmentation	[4]	3.2.4
Position-dependent histogram binning calibration	[1, 4]	3.3.1
Position-dependent logistic and beta calibration	[1, 4]	3.3.2
Derivation of Bayesian confidence calibration	[3]	4.1
Common mathematical context of multiple definitions for regression calibration	[6]	5.2
Parametric regression uncertainty calibration using Gaussian processes	[6]	5.3.2
Joint multivariate regression uncertainty calibration using Gaussian processes	[6]	5.3.2
Covariance estimation of regression uncertainty	[6]	5.4.2
Covariance recalibration of regression uncertainty	[6]	5.4.2
Estimation of object existence over time using the detector's semantic confidence	-	6.3
Integration of semantic confidence calibration to object existence estimation	-	6.3
Estimation of object state using the detector's spatial uncertainty	-	6.4
Integration of spatial uncertainty calibration to object estimation	-	6.4

as methods for uncertainty correction to fit the needs for the task of object detection. We also evaluate our proposed methods on different network architectures and different data sets. The proposed methods for semantic and spatial uncertainty correction are used within Chap. 6 in the context of object tracking to inspect and evaluate their influence on a relevant real-world application. This demonstrates the effectiveness of the proposed methods. Finally, we give a conclusion about our evaluation results, our developed methods, and our findings in Chap. 7.

2 Object Detection and Uncertainty Modeling

Before we start with our main work, we present the basic principle of modern neural networks as well as the architectures for image-based object detection that have been used throughout this work. Moreover, we describe the basic types of uncertainties that are analyzed in common literature. These uncertainties play an important role throughout this work so that we further describe how an object detection model is able to estimate the uncertainty of a prediction. Finally, we give a brief overview of related work that aims to elaborate possible reasons for an unreliable or misaligned uncertainty estimation of modern neural networks.

2.1 Basics of Neural Networks and Image-based Object Detection

In this work, we focus on the image-based object detection process based on neural network architectures. Artificial neural networks are models in the field of machine learning. Machine learning is a concept to enable systems to recognize patterns and characteristics in a known data set. This can subsequently be used to classify new data on the basis of the previously learned patterns and characteristics. For example, in image recognition, images are classified into individual categories. Based on a known data set, the machine learning model is trained to extract the relevant image features to learn a mapping from the input image to the desired category. The trained model can then be used to classify new images that are not part of the training data set.

In this section, we start by introducing the basic concept of a neural network. Furthermore, we describe how to train a neural network so that it is able to apply an appropriate mapping from an input image to the desired output. Finally, we give an overview about the object detection architectures that are based on neural networks and which are used within this work.

2.1.1 Fully-Connected Neural Networks

Basically, a neural network consists of multiple nodes (neurons) that are connected to each other. Each of these connections has a weight that scales the input signal towards the target neuron. Commonly, the neurons are organized into multiple consecutive layers, so that a neural network can be interpreted as a sequence of L consecutive operations (network layers) that take the output of the preceding layer as input to generate a new output. Given an input image $X \in \mathcal{X} = [0,1]^{C \times W \times H}$ of width W, height H, and C channels (e.g., RGB image) with the respective ground-truth label $\bar{Y} \in \mathcal{Y} = \{1, ..., K\}$ with K classes, the whole neural network $d_{\boldsymbol{\theta}}$ serves as a mapping from X to an overall output $Y \in \mathcal{Y}$ (categorical for classification or continuous for regression), so that $d_{\boldsymbol{\theta}} : \mathcal{X} \rightarrow \mathcal{Y}$. In this scope, $\boldsymbol{\theta} \in \Theta$ denote the network parameters. We distinguish between the input layer that consumes the image as input, the intermediate or hidden layers, and the output layer that outputs the overall network prediction $\hat{Y} \in \mathcal{Y}$ which aims to target the real ground-truth label $\bar{Y} \in \mathcal{Y}$.

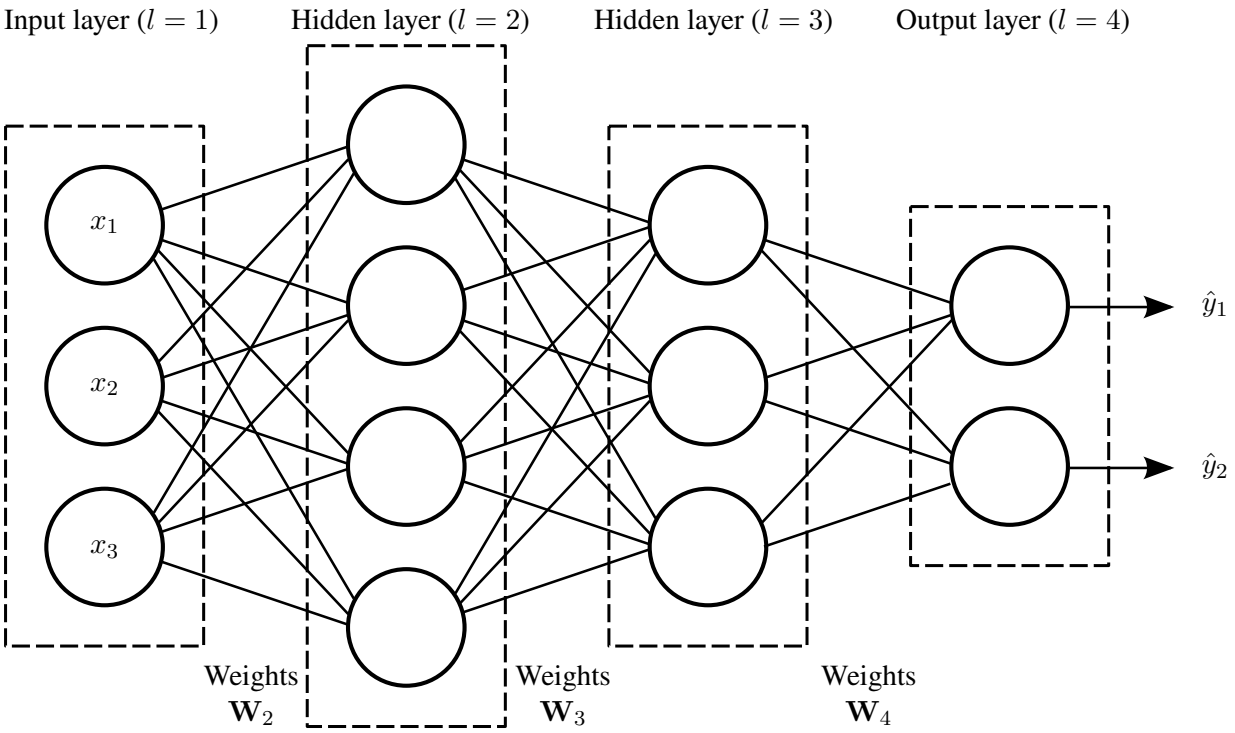


Figure 2.1: Concept of a fully-connected neural network (without bias terms) with two hidden layers, an input layer, and an output layer to obtain the estimate for $\hat{y}_1$, $\hat{y}_2$ based on a given input $\mathbf{x} \in \mathbb{R}^3$.

A basic layer type used in the context of neural networks is the so-called dense or fully-connected layer. The idea is to use multiple nodes or neurons where each neuron is connected to all neurons in the previous layer. Each neuron simply computes a weighted sum over the entire input which is given by the output of the preceding neurons. The fully-connected layer can also be represented by a matrix multiplication between a weight matrix $\mathbf{W}_l \in \mathbb{R}^{M_l \times M_{l-1}}$ and the layer input $\mathbf{x}_l \in \mathbb{R}^{M_{l-1}}$ which is the output of the preceding network layer $l-1$. In this context, M_{l-1} and M_l denote the number of neurons or features of the preceding and the actual layer, respectively. The matrix $\mathbf{W}_l$ holds the weights which are used for the connections between the neurons from layer $l-1$ to l. If the preceding layer is given either by the input image itself or a convolution operation (cf. Sec. 2.1.2), the (3D) input gets flattened to a single vector, so that $M_{l-1} = C_{l-1} \cdot W_{l-1} \cdot H_{l-1}$ with W_{l-1}, H_{l-1}, and C_{l-1} as the width, height, and number of channels within the preceding layer $l-1$. Let $\mathcal{V}_l$ denote the function of layer l. A fully-connected layer applies the transformation $\mathcal{V}_l(\mathbf{x}_l) = \mathbf{W}_l \mathbf{x}_l = \mathbf{x}_{l+1}$ which yields the output vector $\mathbf{x}_{l+1} \in \mathbb{R}^{M_l}$ so that the whole network can be interpreted as a composite function by

$$d_{\boldsymbol{\theta}}(\mathbf{x}_n) = (\mathcal{V}_L \circ \ldots \circ \mathcal{V}_1)(\mathbf{x}_n), \tag{2.1}$$

given a certain sample $\mathbf{x}_N$. A neural network that solely consists of fully-connected layers is denoted a fully-connected neural network. The concept of a such a network is schematically shown in Fig. 2.1.

Activation Functions

A matrix multiplication is a linear transformations of the incoming data. Moreover, the concatenation of several linear operations also results in an overall linear transformation. Therefore, it is not possible to model complex non-linear relationships between input and output space. For this reason, a so-called activation function is usually used which introduces a non-linearity after the linear transformation and between the individual layers. The activation function needs to be differentiable because a neural network is typically trained using the backpropagation algorithm which requires the calculation of the function's derivative. Given an arbitrary function input $t \in \mathbb{R}$, common choices for the intermediate activation functions are the sigmoid given by

$$\Phi(t) = \frac{1}{1 + \exp(-t)}, \tag{2.2}$$

the hyperbolic tangent function (tanh)

$$\tanh(t) = \frac{\exp(2t) - 1}{\exp(2t) + 1}, \tag{2.3}$$

or the Rectified Linear Unit (ReLU) given by $\max(0, t)$ with its derivative set to $\frac{\mathrm{d}\max(0,t)}{\mathrm{d}t} = 0$ for $t = 0$. Furthermore, an activation function is often used after the output layer. For example, in binary classification, if output scores in the $[0, 1]$ interval are required, a common choice is the sigmoid function. In multiclass classification with K classes and $\mathbf{t} \in \mathbb{R}^K$, the softmax function given by

$$\Phi_{\mathrm{SM}}(\mathbf{t})_k = \frac{\exp(t_k)}{\sum_{k'=1}^{K} \exp(t_{k'})} \tag{2.4}$$

is a common choice to obtain predictions whose outputs sum up to 1, i.e., $\sum_{k=1}^{K} \hat{y}_k = 1$. In both cases, the output can be interpreted as parameters for a Bernoulli or categorical distribution in binary or multiclass classification, respectively, which allows for a probabilistic interpretation of the network output. The concept of a single neuron within a fully-connected neural network using a subsequent activation function is schematically shown in Fig. 2.2.

Training of the Network Weights[1]

According to the principle of Empirical Risk Minimization (ERM), it is not possible to know the true real-world distribution of the data (e.g., the distribution of all possible images) so that we can not exactly determine the performance of our algorithm [50]. Instead, we are restricted to a certain set of training data which is drawn from the true data distribution and on which we can quantify the model performance (the empirical risk). Let $\mathcal{D} = \{(\mathbf{x}_n, \bar{y}_n)\}_{n=1}^{N}$ denote a data set of size N with images $\mathbf{x}_n \in \mathcal{X}$ and the known target labels $\bar{y}_n \in \mathcal{Y}$. The training set is generated by a probability distribution $\mathcal{P}$ over $\mathcal{X}$. Furthermore, we assert that a correct labeling function u^* exists so that $u^* : \mathcal{X} \to \mathcal{Y}$. The task for a learning algorithm is to output

[1] The descriptions for training a neural network are partly adapted from previous works [48, 49].

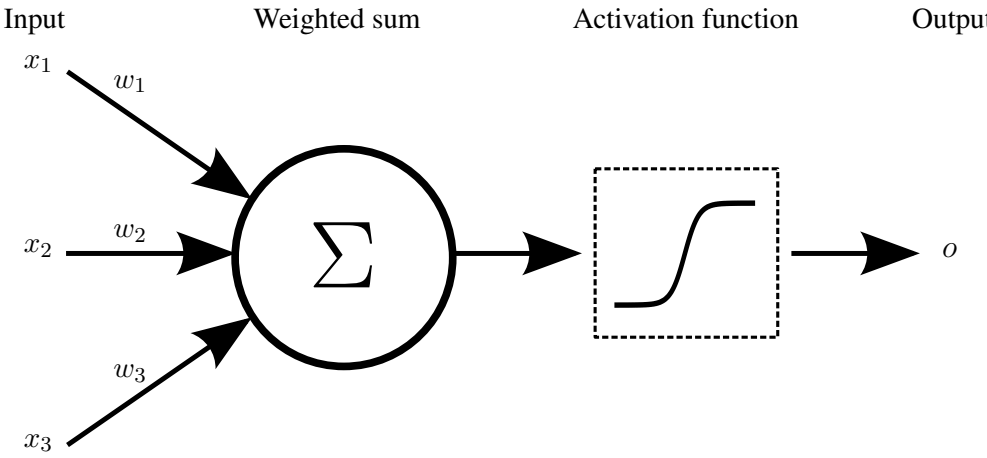


Figure 2.2: Concept of a single neuron with a subsequent non-linear activation function to generate an output $o \in \mathbb{R}$. The neuron takes either the network input or the preceding layer output and computes a weighted sum over the complete input vector. The result is given to the activation function and then passed on to the next network layer.

a prediction rule $d_{\mathcal{D}} : \mathcal{X} \to \mathcal{Y}$ based on the training set $\mathcal{D}$ to classify the images. The error $\mathcal{L}_{\mathcal{P},u^*}$ of the prediction rule is given by

$$\mathcal{L}_{\mathcal{P},u^*}(d_{\mathcal{D}}) = \mathbb{P}_{\mathbf{X}\sim\mathcal{P}}\big[d_{\mathcal{D}}(\mathbf{x}) \neq u^*(\mathbf{x})\big]. \tag{2.5}$$

Since the distribution $\mathcal{P}$ as well as the true labeling function u^* are unknown, the learner needs to find a prediction rule $d_{\mathcal{D}}$ that minimizes the prediction error on the given data set $\mathcal{D}$. It is only possible to observe the true labeling function u^* for the samples that are present in the data set $\mathcal{D}$ as we know the true mapping from the input $\mathbf{x}$ to the desired outcome $\bar{y}$. Thus, the training error or empirical error of a learner is given by

$$\mathcal{L}_{\mathcal{D}}(d_{\mathcal{D}}) = \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}(d_{\mathcal{D}}(\mathbf{x}_n) \neq u^*(\mathbf{x}_n)), \tag{2.6}$$

where $\mathbb{1}(\cdot)$ is the indicator function which evaluates to 1 if the argument is true and 0 otherwise.

Maximum Likelihood Estimation (MLE) is a method to learn the parameters $\boldsymbol{\theta}$ that are used to model the underlying probability distribution $\mathcal{P}$ by an approximate distribution $\mathcal{P}_{\boldsymbol{\theta}}$ parameterized by $\boldsymbol{\theta}$ to construct an appropriate prediction rule $d_{\boldsymbol{\theta}}$ [51]. The maximum likelihood estimation is a minimization technique for ERM using the log loss on $\mathcal{P}_{\boldsymbol{\theta}}$ which is given by

$$-\log\big(\mathcal{P}_{\boldsymbol{\theta}}(\mathbf{x})\big), \tag{2.7}$$

with parameters $\boldsymbol{\theta}$ and input data $\mathbf{x}$. The equation (2.7) is also known as the Negative Log Likelihood (NLL).

Using this definition, maximizing the log likelihood is equivalent to minimizing the empirical risk, since

$$\underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{n=1}^{N} -\log\big(\mathcal{P}_{\boldsymbol{\theta}}(\mathbf{x}_n)\big) = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{n=1}^{N} \log\big(\mathcal{P}_{\boldsymbol{\theta}}(\mathbf{x}_n)\big), \tag{2.8}$$

and it can be shown that the true risk of parameter $\boldsymbol{\theta}$ is given by

$$\mathbb{E}_{\mathbf{X}\sim\mathcal{P}}\Big[-\log\big(\mathcal{P}_{\boldsymbol{\theta}}(\mathbf{x})\big)\Big] = D_{\mathrm{KL}}\big(\mathcal{P}(\mathbf{x})||\mathcal{P}_{\boldsymbol{\theta}}(\mathbf{x})\big) + \mathcal{H}\big(\mathcal{P}(\mathbf{x})\big), \tag{2.9}$$

with $D_{\mathrm{KL}}\big(\mathcal{P}(\mathbf{x})||\mathcal{P}_{\boldsymbol{\theta}}(\mathbf{x})\big)$ as the Kullback-Leibler divergence [52] between true data distribution $\mathcal{P}$ and the estimated one $\mathcal{P}_{\boldsymbol{\theta}}$, and the Shannon entropy $\mathcal{H}\big(\mathcal{P}(\mathbf{x})\big)$ [53] of the true data distribution.

Now, given a neural network d with $\boldsymbol{\theta}$ parameters, the objective of MLE is to obtain a $\hat{\boldsymbol{\theta}}$ that maximizes the likelihood of observing the data. Thus, during network training, it is required to pass each input image through the network so that the network is requested to output a prediction $\hat{y}_n = d_{\boldsymbol{\theta}}(\mathbf{x}_n)$. Afterwards, the predicted output $\hat{y}_n$ is evaluated against the ground-truth label $\bar{y}_n$. A cost or loss function $\mathcal{L}(\boldsymbol{\theta})$ such as the NLL in (2.7) is used to implement the maximum likelihood estimation. In a binary classification setting ($\mathcal{Y} = \{0, 1\}$), the likelihood function is a Bernoulli distribution so that the NLL is given by

$$\mathcal{L}(\boldsymbol{\theta}) = -\sum_{n=1}^{N} \bar{y}_n \log(\hat{y}_n) + (1 - \bar{y}_n)\log(1 - \hat{y}_n). \tag{2.10}$$

In order to minimize the loss function $\mathcal{L}(\boldsymbol{\theta})$, the Stochastic Gradient Descent (SGD) algorithm is used to update the network parameters $\boldsymbol{\theta}$. An initial guess for the network parameters $\boldsymbol{\theta}$ is used as a starting point to iteratively descend in the direction of the minimum. The direction of the update is determined by the gradient of the network components w.r.t. the parameters $\boldsymbol{\theta}$. Thus, the update for a weight matrix $\mathbf{W}_l$ at layer l is calculated by

$$\mathbf{W}_l = \mathbf{W}_l - \eta \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathbf{W}_l}, \tag{2.11}$$

with $\eta \in \mathbb{R}_{>0}$ as the learning rate which controls the strength of the parameter update. Commonly, the weight update is not applied using the whole data set $\mathcal{D}$. Instead, the training data is grouped into batches which are small subsets of $\mathcal{D}$. During network training, the loss is repeatedly computed over all batches that are present in the training set. In this way, the optimization is computationally more efficient and it may help to overcome local minima in the loss function [54, 55].

The gradients, which are used by the SGD algorithm, are obtained using backpropagation [56]. Since the neural network is grouped into several consecutive layers, the output and thus the loss $\mathcal{L}(\boldsymbol{\theta})$ are composite functions of multiple consecutive layer operations $\big($cf. equation (2.1)$\big)$. Thus, for the intermediate network

layers, it is required to obtain the gradient using the chain rule

$$\frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathbf{W}_l} = \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}\mathcal{V}_L}\frac{\mathrm{d}\mathcal{V}_L}{\mathrm{d}\mathcal{V}_{L-1}}\cdots\frac{\mathrm{d}\mathcal{V}_{l+1}}{\mathrm{d}\mathcal{V}_l}\frac{\mathrm{d}\mathcal{V}_l}{\mathrm{d}\mathbf{W}_l}, \tag{2.12}$$

to finally yield the gradient for the weight update. This process is repeated several times to finally approximate the ground-truth data distribution $f_{\bar{Y}}(\bar{y}|\mathbf{x})$ by an estimated network distribution $f_{\hat{Y}}(\hat{y}|\mathbf{x})$.

Regularization

The minimization of the empirical risk may lead to overfitting, i.e., the estimator $d_{\mathcal{D}}$ works well on the given training data set $\mathcal{D}$ but fails to estimate the correct labels on unseen data. Weight decay [57] is a regularization technique that adds an additional penalty to the loss term which penalizes large layer weights. Given the loss function w.r.t. a dedicated weight matrix $\mathbf{W}_l$, the equation for the loss including the weight decay is given by

$$\bar{\mathcal{L}}(\mathbf{W}_l) = \mathcal{L}(\mathbf{W}_l) + \frac{\eta'}{2}||\mathbf{W}_l||_2^2, \tag{2.13}$$

where the parameter $\eta' \in \mathbb{R}_{>0}$ controls the strength of the weight decay. Using this technique for regularization, the weights of a layer are decreased in each iteration by a constant factor which reduces the model capacity and helps to prevent overfitting.

In addition, dropout has originally been proposed as a regularization technique during network training. When applying dropout, single network weights are randomly deactivated so that the network is requested to apply a mapping without these specific connections. This behavior also aims to reduce model capacity and strengthen the remaining network connections. Moreover, the authors in [58] showed that dropout can also be used during inference to quantify the uncertainty of the network about a specific prediction. During inference, the same input is passed multiple times through the network. In each forward pass, dropout is applied on different randomly chosen network connections. This yields in multiple predictions for the same input that span a sample distribution as the network output. The higher the variance of this output distribution, the higher the network's uncertainty about the current prediction. This process is known as Monte-Carlo dropout [58].

Finally, the authors in [59] proposed batch normalization as a technique to normalize the output after each layer l. This method aims to improve the network performance by minimizing the distribution shifts (internal covariate shift) after the hidden layer's activation functions. Given the layer input $\mathbf{x}_l \in \mathbb{R}^K$ with K dimensions, the batch normalization is calculated by

$$x_{l,k}^* = \frac{x_{l,k} - \mu_{\mathcal{B},k}}{\sqrt{\sigma_{\mathcal{B},k}^2 + \epsilon}}, \tag{2.14}$$

where $\mu_{\mathcal{B},k}$ and $\sigma_{\mathcal{B},k}^2$ are the mean and variance within batch $\mathcal{B}$ and dimension k, respectively, with $\epsilon \in \mathbb{R}$ as a small offset to increase numerical stability.

The authors in [59] note that a normalization of each layer input may affect its representational power. The authors propose to use additional parameters so that, during network training, it is possible to revert or change the normalization, e.g., to obtain an identity transform. Therefore, after the batch normalization step in (2.14), a shifting and rescaling of the input is applied by

$$x_{l,k}^{**} = \gamma_k x_{l,k}^{*} + \delta_k, \tag{2.15}$$

with scaling parameter $\gamma_k \in \mathbb{R}_{>0}$ and shift parameter $\delta_k \in \mathbb{R}$ for all $k \in \{1, \dots K\}$.

2.1.2 Convolutional Neural Network

A Convolutional Neural Network (CNN) is a network that utilizes at least one convolution operation as an alternative type to the fully-connected layer in order to process the input data [60]. A convolution is a linear operation on two functions $f : \mathbb{R} \to \mathbb{R}$ and $g : \mathbb{R} \to \mathbb{R}$ defined by

$$(f * g)(t) = \int_{-\infty}^{+\infty} f(\tau) g(t - \tau) \mathrm{d}\tau, \tag{2.16}$$

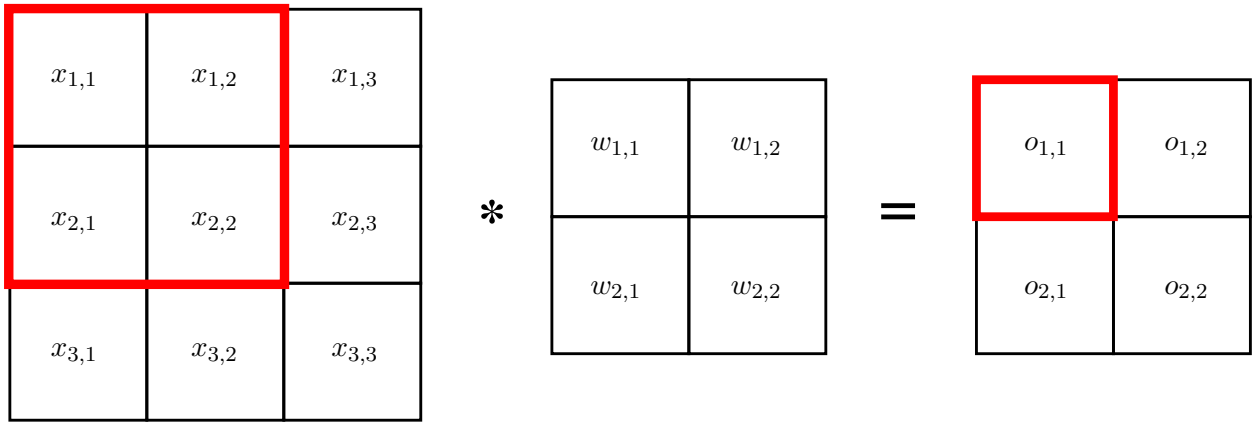
or, in the discrete case with $f : \mathbb{Z} \to \mathbb{R}$ and $g : \mathbb{Z} \to \mathbb{R}$, by

$$(f * g)(t) = \sum_{\tau \in \mathbb{Z}} f(\tau) g(t - \tau). \tag{2.17}$$

In image processing, the convolution describes the process of calculating a weighted sum of neighboring pixels using a filter mask within the original image. The weights are the coefficients of the filter mask. The filter mask is moved successively over all image positions to apply the filtering to the complete image and to get feature representations for all image regions. This process is schematically shown in Fig. 2.3. More formally, given the layer input $\mathbf{x}_l \in \mathbb{R}^{C_{l-1} \times W_{l-1} \times H_{l-1}}$ of width W_{l-1}, height H_{l-1}, and C_{l-1} channels (obtained by the preceding layer $l-1$), the discrete convolution operation is defined by

$$x_{l+1}^{(c_l, i, j)} = \sum_{c_{l-1}=1}^{C_{l-1}} \sum_{w=1}^{W_k} \sum_{h=1}^{H_k} x_l^{(c_{l-1}, i+w, j+h)} \cdot w_l^{(c_l, c_{l-1}, w, h)} \tag{2.18}$$

given the filter mask $\mathbf{W}_l \in \mathbb{R}^{C_l \times C_{l-1} \times W_k \times H_k}$ with filter width W_k, filter height H_k, and C_l channels. Note that it is only possible to shift the filter kernel so far until it reaches the image or feature map boundary. A mitigation of this problem is called padding where missing pixel values are padded by a certain scheme, e.g., by adding zeros to the missing pixel positions (zero padding). The application of the convolutional operation is a common technique used in computer vision. The convolution filters are used to obtain feature representations (e.g., detected edges) to further process the input image. The advantage of applying a convolution is that the layer is equivariant to translation, i.e., the filter mask is applied to all image regions with the same filter weights. In the context of CNN, it is possible to learn the filter mask weights $\mathbf{W}_l$ by



(a) Input image or feature map (b) Filter mask (c) Output feature map

Figure 2.3: Concept of the convolution operation which is applied on an input image or feature map (a) with a certain filter mask (b) to extract a new feature map (c). This result of the convolution can represent certain image features such as edges or other structural information.

applying stochastic gradient descent using backpropagation algorithm (cf. next section). Note that, since the convolution is a linear operation similar to the fully-connected layer, a subsequent activation function is also required for a CNN after each layer. In image classification, the output of a CNN is commonly a vector that describes the estimated class probabilities given a certain input image. In order to calculate the output vector, the results after the convolutional layers are commonly passed to a fully-connected layer which generates the final network output.

2.1.3 Architectures for Object Detection

In contrast to simple image recognition, where the label of the whole image is of interest, the task of object detection is the joint task of classification (object type) and regression (object position). Thus, a neural network architecture for object detection $d_{\boldsymbol{\theta}}$ needs to solve both tasks simultaneously given an input image $\mathbf{x}_n$. In the following, we present the basic architectures of modern neural-network based object detectors.

Feature Extraction and Object Candidates

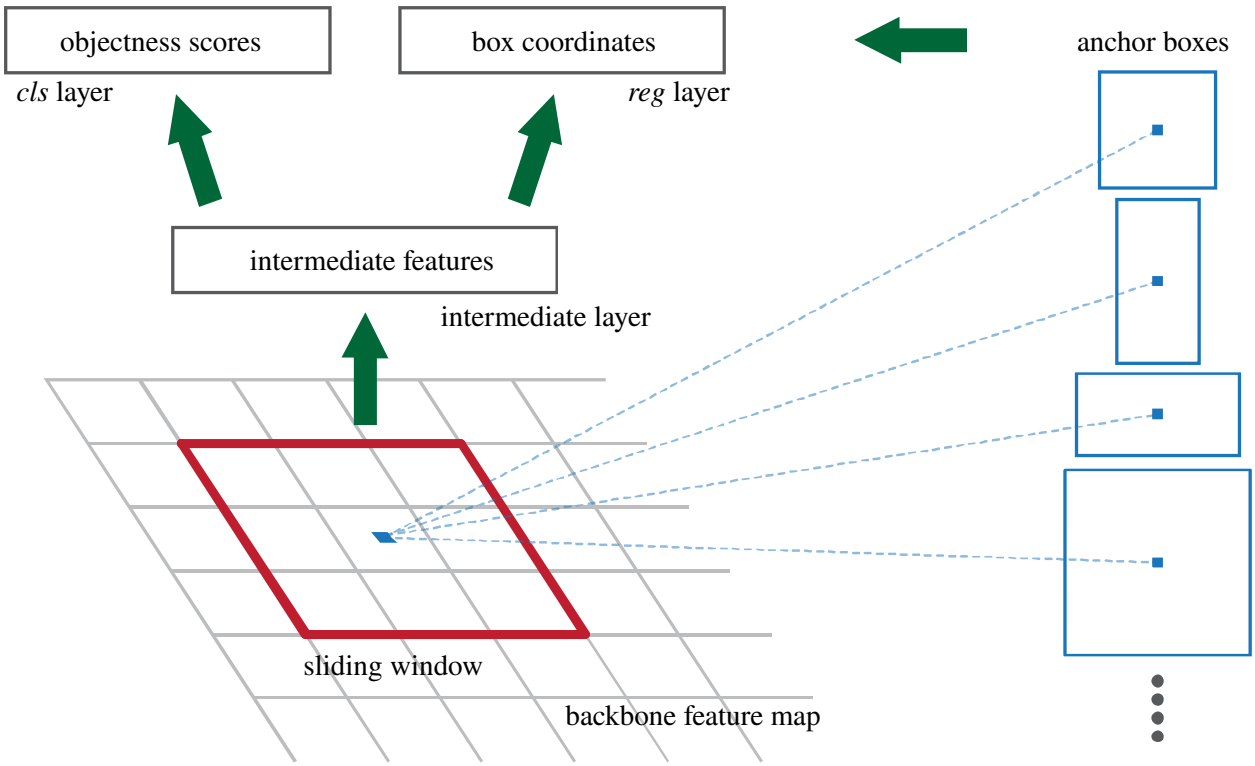
Each of the object detector algorithms are based on a backbone network to extract relevant image features which are further processed for object classification and position/shape estimation. The backbone is usually a CNN that processes an image and generates intermediate feature representations of the input using its convolution operations. This network is commonly adapted from classification and pretrained on a classification data set [61]. Thus, passing the input image $\mathbf{x}_n$ through the backbone network $d_{\boldsymbol{\theta}_B}$ results in a feature representation $d_{\boldsymbol{\theta}_B}(\mathbf{x}_n) = \hat{y}_B \in \mathbb{R}^{C_B \times W_B \times H_B}$ of the input image with C_B feature maps of width W_B and height H_B. At this point, it is necessary to distinguish between a two-stage and a single-stage object detector. While a single-stage detector directly works with the backbone network, a two-stage detector uses an addi-

tional step to generate prior assumptions about possible object candidates. In a two-stage architecture, the feature maps of the backbone network are passed to a subsequent Region Proposal Network (RPN) denoted by $d_{\boldsymbol{\theta}_{\text{RPN}}}$ [61]. The RPN itself is a small CNN that outputs a list of U object proposals by sliding its convolutional network on the feature maps. These object proposals are initial guesses for possible object candidates. For the computation of the object proposals, so-called anchor boxes are used for each location of the sliding window. The anchor boxes can be interpreted as priors for the object proposals, as they are used at each sliding window location of the RPN with fixed object locations and fixed aspect ratios. Subsequently, the RPN uses two final fully-connected layer to compute an objectness score as well as the final coordinates for the object proposals. The objectness score $\hat{p}_u$ represents the belief of the RPN about the presence or absence of an object within a certain anchor. This score is used to filter the object proposals so that only proposals with an objectness score above a certain threshold are passed through the network. Furthermore, the RPN outputs the center coordinates $c_{x,u}, c_{y,u}$ as well as the width and height w_u, h_u of the proposal boxes. Thus, the output of the RPN is given by $(\hat{p}_u, c_{x,u}, c_{y,u}, w_u, h_u)^\top$ for all $u \in \{1, \dots, U\}$.

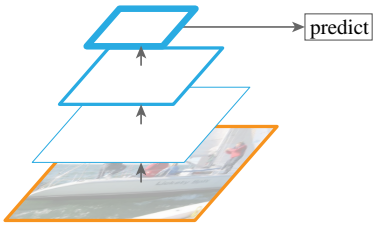
A drawback of the standard RPN architecture is that it only consumes the final output of the backbone network which might represent features at a single scale [62]. In common image recognition systems, so-called feature pyramids are used to extract image features at different scales, i.e, features that represent either low-level (local) structures (e.g., single window, arm, leg) or high-level (global) structures (e.g., building, car, human body). Recently, the authors in [62] proposed a Feature Pyramid Network (FPN) that is designed to mitigate the limitations of a RPN by extracting the feature maps of the backbone network at different scales to construct a feature pyramid. Furthermore, the different layers/scales of the FPN are connected to each other to obtain a more flexible and computationally efficient architecture for object proposal generation. The concept of the RPN as well as the FPN architectures are schematically shown in Fig. 2.4.

Two-Stage Detector: Faster R-CNN

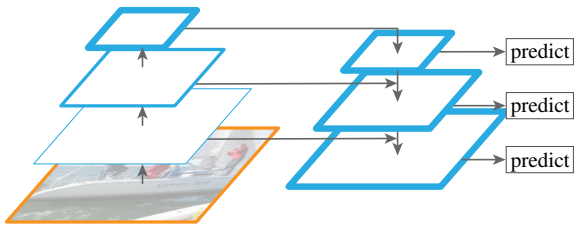
Previously, we mentioned the distinction of modern neural-network based object detection algorithms into two-stage architectures such as Faster R-CNN [61] and single-stage architectures such as Single-Shot detector [24], YOLO [25], and RetinaNet [9]. A two-stage detector utilizes the previously described object proposal stage using a RPN network. Furthermore, it is also possible to use a preceding FPN before the RPN for further feature processing to improve the quality of the object proposals [62]. The object proposals as well as the feature maps $d_{\boldsymbol{\theta}_B}(\mathbf{x}_n) = \hat{y}_B$ of the backbone network are passed to a fully-connected classification network $d_{\boldsymbol{\theta}_{\text{CLS}}}$ and to a fully-connected refinement network for the bounding boxes $d_{\boldsymbol{\theta}_{\text{BOX}}}$ to generate the final object predictions $\hat{y}_n$ with the according bounding box positions $\hat{\mathbf{r}}_n \in \mathcal{R}$. The object proposals are used to crop the relevant features at the proposed object locations from the backbone feature maps $\hat{y}_B$. The classification head $d_{\boldsymbol{\theta}_{\text{CLS}}}$ takes these features as input to generate the final prediction for the object class which might be one out of K classes. The final network output is passed to the softmax activation function to obtain a probabilistic interpretation of the class probabilities for a single object. For the estimation of the final bounding box of an object, the refinement head $d_{\boldsymbol{\theta}_{\text{BOX}}}$ works similarly. The refinement network takes the relevant features from $\hat{y}_B$ at the proposed object locations and predicts an offset for the proposed object locations $\Delta c_x, \Delta c_y$ and



(i) Concept of the Region Proposal Network (RPN) using a sliding window with different predefined anchor boxes for each position within the output feature maps [61, p. 3, Fig. 1].



(ii) Basic structure of a Region Proposal Network (RPN) that uses the final output of the backbone network [62, p. 1, Fig. 1(b)].



(iii) Feature Pyramid Network (FPN) that also the intermediate features of the backbone to estimate proposals at different scales [62, p. 1, Fig. 1(d)].

Figure 2.4: Concept of the Region Proposal Network (RPN) and the Feature Pyramid Network (FPN) used for object detection. For object detection using neural networks, it is necessary to make an initial assumption about possible object candidates and their locations [61, 9, 62]. Initially, a RPN has been proposed by [61] to output object proposals by using a sliding window over the backbone feature maps with predefined anchor boxes (top). However, the drawback of this approach is that it only utilizes the features at the output of the backbone network (left bottom) which might only work well at a single scale [62]. Therefore, the authors in [62] proposed a FPN to mitigate this problem by utilizing the intermediate backbone features at different scales.

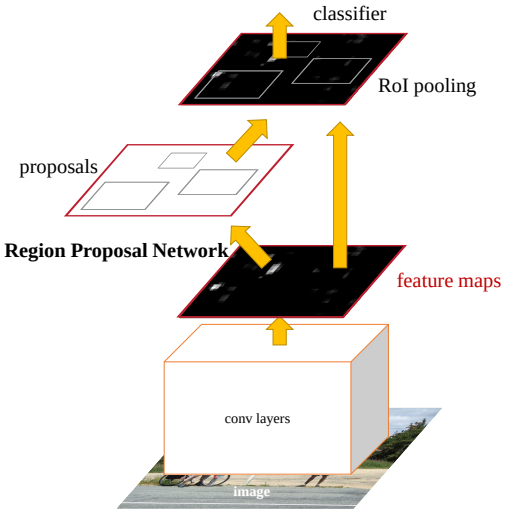


Figure 2.5: Concept of the Faster R-CNN object detection architecture [61, p. 3, Fig. 2]. A basic backbone network (denotes by "conv layers") is used to generate feature representations of the input image. Building on top of these features, the region proposal network generates multiple object proposals with possible object candidates. In the final step, the final pooling layer (denoted by "RoI pooling") discards all proposals that are classified as background. Furthermore, the final stage applies a refinement of the bounding boxes that have not been classified as background.

for the proposed width and height $\Delta w, \Delta h$. Thus, the two-stage detection architecture is a concatenation of the backbone network, the proposal stage, and the detection head, so that the final object predictions are obtained by $(d_{\boldsymbol{\theta}_B} \circ d_{\boldsymbol{\theta}_{\text{RPN}}} \circ d_{\boldsymbol{\theta}_{\text{CLS}}})(\mathbf{x}_n)$ for the labels of the objects and $(d_{\boldsymbol{\theta}_B} \circ d_{\boldsymbol{\theta}_{\text{RPN}}} \circ d_{\boldsymbol{\theta}_{\text{REG}}})(\mathbf{x}_n)$ for the object positions given an input image $\mathbf{x}_n$. During network training, the backbone network, the proposal network as well as the detection head are jointly optimized using the classification and refinement loss of the proposal stage in conjunction with the classification and refinement loss of the detection head. The concept of the Faster R-CNN detection architecture is schematically shown in Fig. 2.5.

Single-Stage Detector: RetinaNet

A single-stage detection architecture such as a RetinaNet [9] follows a similar concept of using a pretrained backbone network to generate an intermediate feature representation of the input image. As opposed to the previously described two-stage object detection approach, a single-stage detector does not utilize an additional proposal stage. Instead, a single-stage architecture directly works with the anchor boxes that we already know from the RPN architecture. The anchor boxes are placed with fixed object locations and fixed aspect ratios over the whole image output and serve as priors for possible object candidates. The single-stage object detector seeks to directly learn an appropriate scaling and shifting of these boxes to generate the final object predictions. Thus, an additional proposal stage is not necessary. The anchor boxes are connected to certain feature maps of intermediate layers within the backbone network to extract image features at different scales. Furthermore, the RetinaNet architecture uses a FPN before scaling and shifting the anchor boxes to

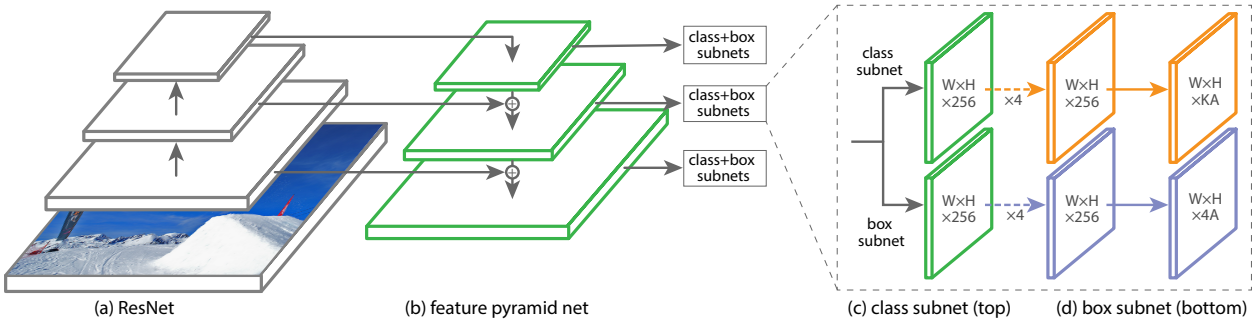


Figure 2.6: Concept of the RetinaNet object detection architecture [9, p. 5, Fig. 3]. Similar to a two-stage detector, the RetinaNet uses a backbone network (left) to generate intermediate feature representations of the input image. A feature pyramid network extracts these feature representations at predefined anchor locations and seeks to detect possible correlations between features of different scale. Finally, the last refinement stage classifies the anchors based on the intermediate features and applies a bounding box refinement of the anchor boxes to match the estimated objects within an image.

improve the feature extraction process [62]. The output layers $d_{\boldsymbol{\theta}_{\text{CLS}}}$ and $d_{\boldsymbol{\theta}_{\text{BOX}}}$ are similar as to the ones used within two-stage detectors as they also estimate the final object class as well as the final object position and size. The concept of the RetinaNet architecture is schematically shown in Fig. 2.6.

2.2 Different Types of Uncertainty

The formerly presented object detection algorithms are deep learning techniques that are able to learn and recognize features and patterns in the given input images. Although many machine learning and deep learning algorithms are designed to output deterministic estimates, such a process is always subject to certain uncertainties during inference [39, 63]. In this scope, the authors in [63] identified 3 different reasons why a machine learner is subject to uncertainty:

1. The dependency between input space and the respective true outcome (e.g., input image and its associate classification label) may be of stochastic nature [63]. For example, given two overlapping distributions within the same input space, the translation from input space to the output, which represents the respective distribution label, is not deterministic. This is known as the aleatoric data uncertainty.

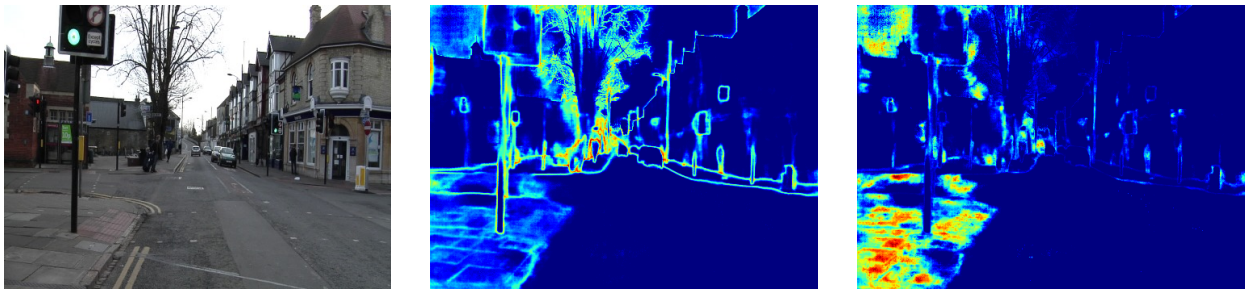
2. Before applied to real-world applications, a machine learning algorithm needs a dedicated training phase to learn the translation from input to output space given a dedicated training set. However, the training set is always a random sample drawn from the data generation process. This leaves a lack of knowledge during application which is referred to as approximation uncertainty by [63].

3. A machine learning or deep learning model might be misspecified [63], i.e., it might not be able to correctly capture all dependencies between input and output space. For example, a neural network

with low capacity might fail in a classification of input images, whereas a network architecture with high capacity might succeed. The authors in [63] denote this as model uncertainty [63].

These uncertainties are commonly divided into epistemic model and aleatoric data uncertainty [64, 39, 63]. The difference between these types of uncertainty is shown in Fig. 2.7.

Epistemic uncertainty represents the lack of knowledge inherent in the model due to a low amount of training data or due to model misspecification [39, 63]. This uncertainty is commonly reducible given more training data [39] or by an improved model specification [63]. In the context of neural networks, epistemic uncertainty is not directly observable, i.e., we need advanced techniques to access this type of uncertainty. For this reason, Bayesian neural networks (BNN) have been introduced to mitigate this problem [65, 66]. The idea of a BNN is to place probability distributions over the (commonly deterministic) network weights to yield a probability distribution as the network output. However, since neural networks exhibit a complex structure and work with intermediate non-linearities, the output distribution is analytically not tractable [58]. Therefore, recent works have introduced techniques for an approximation of BNNs. The authors in [67] use an ensemble of neural networks during inference with the same architecture to yield a sample distribution as output given a single input. The authors in [58] show that dropout, a common regularization technique where single neurons get randomly deactivated within the network training, can also be used as an ensemble approximation during network inference given new data. Another technique is Stochastic Variational Inference (SVI) [68] where the network weights are replaced by variational distributions of known parametric form (e.g., Gaussian). Using SVI, it is possible to approximate the variational distribution parameters during network training. During inference, we can sample from these variational distributions to get multiple parameter sets for the neural network. However, a known problem of SVI is that is does not scale well to large neural networks [68]. All of these techniques approximate a BNN by yielding a sample distribution for the network output given a certain input. Recently, the authors in [69] derived a sampling-free technique to yield epistemic uncertainty



(i) Input image. (ii) Aleatoric uncertainty. (iii) Epistemic uncertainty.

Figure 2.7: Difference between aleatoric and epistemic uncertainty demonstrated at the task of semantic segmentation [39, p. 2, Fig. 1]. We observe that aleatoric data uncertainty occurs on segment boundaries or image regions with blurry contours, e.g., the top of the tree in the image center. In contrast, the epistemic uncertainty represents the intrinsic model uncertainty that might occur at different locations that are not well-known by the segmentation model.

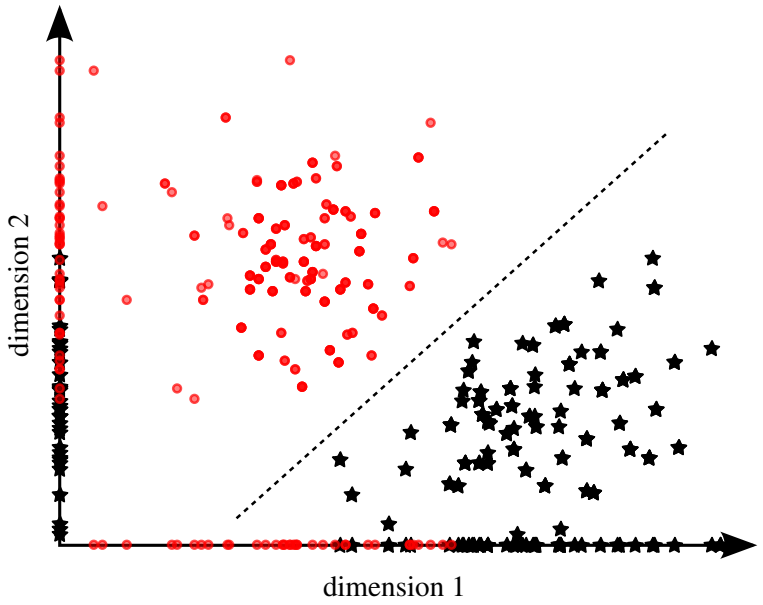


Figure 2.8: Given two non-overlapping distributions in a 2-D feature space, the distributions are separable and we have no aleatoric uncertainty. However, if projected only to a single dimension and provided to a machine learning model, the distributions might overlap which results in regions with aleatoric uncertainty.

by using Gaussian error propagation as an approximation to BNNs. In this way, it is possible to represent the intermediate as well as the final uncertainty as Gaussian distributions that are propagated through the whole network architecture [69]. Using all of these techniques, it is possible to obtain epistemic uncertainty of a neural network. If transferred to object detection architectures (cf. Sec. 2.1), we face a problem of BNNs during the inference of objects. A `Faster R-CNN` as well as a `RetinaNet` both work with proposal or anchor boxes before the final refinement stage. If we now apply the sampling techniques as described above, we obtain multiple bounding box estimates for the same image. Furthermore, the assignment of the proposal/anchor boxes to individual objects might change with each forward pass in the sampling process. Thus, the authors in [70] and [16] use clustering techniques to group the predicted objects after sampling. Consequently, it is also possible to obtain epistemic uncertainty within the detection process.

Aleatoric uncertainty represents the uncertainty inherent in the data. This kind of uncertainty can not be explained away given more data [39]. However, the authors in [63] argue that aleatoric uncertainty might also result from a misspecification of the problem setting. For example, given two non-overlapping distributions in a 2-D feature space, the distributions are separable and we have no aleatoric uncertainty. However, if projected only to a single dimension and provided to a machine learning model, the distributions might overlap in this dimension which results in a region with aleatoric uncertainty. This is schematically shown in Fig. 2.8. This example shall illustrate that aleatoric uncertainty might be mitigated given enough information to a machine learning algorithm [63]. The advantage of working with aleatoric uncertainty is that we can train a neural network to directly output an estimate of this kind of uncertainty. Since we are working with object detection algorithms, we further need to distinguish between semantic label and spatial position uncertainty.

2.2.1 Modeling of Semantic Confidence

Semantic confidence describes the belief of the learning model about the object category of a single object. A neural network commonly estimates a score $Z_k \in \mathbb{R}$ (logit) for each class k in the set of all classes $\mathcal{Y} = \{1, ..., K\}$. Afterwards, the logit of a neural network can be converted to a confidence score $\hat{P}_k \in [0, 1]$ for each class $k \in \{1, ..., K\}$ using the softmax function $\Phi_{\text{SM}}(Z)_k$. The advantage of using the softmax function is that it yields confidence scores which sum up to 1 and which, in turn, can be used to construct a categorical probability distribution targeting the predicted class for each object. In contrast, for binary classification with $\mathcal{Y} = \{0, 1\}$, a neural network only predicts a single logit Z which is converted to a confidence score using the sigmoid function $\Phi(Z)$. Similar to the multivariate case, this confidence score is used to construct a Bernoulli distribution which reflects the probability for an outcome belonging either to class 0 or 1, respectively. In both cases, we treat the network output as a probabilistic forecast which allows to derive the uncertainties directly by the network output.

2.2.2 Modeling of Spatial Uncertainty

The spatial uncertainty reflects the uncertainty during the prediction of an object position and shape within a single image. For position and shape estimation, common object detection architectures use anchor or proposal boxes as the basis for inference (cf. Sec. 2.1.3). A neural network learns to rescale and shift these boxes during training to match possible objects within an image. This architecture has the advantage that a neural network only needs to learn a rescaling and shifting of these boxes which is a simple regression task. Let $\mathbf{X} \in \mathcal{X}$ denote the input to an object detector within the input set $\mathcal{X}$ and let $\mathcal{D}$ denote a training data set of size N which is used to train the parameters $\boldsymbol{\theta}$ of a neural network. Within the training data set, the variable $\bar{\mathbf{r}} \in \mathcal{R}$ represents the ground-truth bounding box information of individual objects of dimension L in the space $\mathcal{R}$ consisting of the position, width, and height information. A standard regression model interprets the network output $\hat{\mathbf{R}}$ as multiple independent normal distributions for each bounding box dimension with fixed variance, so that $\hat{\mathbf{R}} \sim \mathcal{N}(\boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}}, \boldsymbol{\Sigma})$, with $\boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}} \in \mathcal{R}$ as the predicted mean vector and $\boldsymbol{\Sigma} = \text{diag}(\sigma_1^2, \ldots, \sigma_L^2)$ as the diagonal covariance matrix with variances $\sigma_1^2, \ldots, \sigma_L^2 \in \mathbb{R}_{>0}$. The variances are commonly treated as fixed constants, so that $\boldsymbol{\Sigma} = \sigma^2 \mathbf{I}$ with σ^2 as a common variance (e.g., 1) and $\mathbf{I}$ as the identity matrix. Since the variances are treated as constants, they are neglected within standard regression applications. A regression model is trained using the NLL of the model given the training set, so that the loss is defined by

$$\mathcal{L}(\boldsymbol{\theta}) = -\log\left(\prod_{n=1}^{N} f_{\hat{\mathbf{R}}}(\bar{\mathbf{r}}_n|\mathbf{x}_n, \boldsymbol{\theta})\right) \tag{2.19}$$

$$\propto \sum_{n=1}^{N}\sum_{l=1}^{L}\left(\bar{r}_{n,l} - \mu_{\hat{R}_l|\mathbf{x}_n}\right)^2, \tag{2.20}$$

which is also known as the squared error loss function.

A drawback of this interpretation is that the uncertainty, i.e., the variance, is not considered during model training and inference. Thus, no information about the spatial uncertainty is available. For this reason, recent work has reformulated this loss by also considering the variance as a function of the input data. Therefore, the covariance matrix $\mathbf{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}} \in \mathbb{R}_{>0}^{L\times L}$ consisting of the independent variances $\sigma^2_{\hat{R}_l|\mathbf{X}}$ for each dimension $l \in \{1,\dots,L\}$ is also modeled by the regression network for each input $\mathbf{x}_n$, so that $\mathbf{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}} = \mathrm{diag}\big(\sigma^2_{\hat{R}_1|\mathbf{X}},\dots,\sigma^2_{\hat{R}_l|\mathbf{X}}\big)$ [39]. Thus, the NLL extends to

$$\mathcal{L}(\boldsymbol{\theta}) = -\log\left(\prod_{n=1}^{N} f_{\hat{\mathbf{R}}}(\bar{\mathbf{r}}_n|\mathbf{x}_n,\boldsymbol{\theta})\right) \tag{2.21}$$

$$= -\log\left(\prod_{n=1}^{N}\prod_{l=1}^{L} \frac{1}{\sqrt{2\pi\sigma^2_{\hat{R}_l|\mathbf{x}_n}}} \exp\left[\frac{-\big(\bar{r}_{n,l}-\mu_{\hat{R}_l|\mathbf{x}_n}\big)^2}{2\sigma^2_{\hat{R}_l|\mathbf{x}_n}}\right]\right) \tag{2.22}$$

$$\propto \sum_{n=1}^{N}\sum_{l=1}^{L} \frac{1}{2}\sigma^{-2}_{\hat{R}_l|\mathbf{x}_n}\Big(\bar{r}_n-\mu_{\hat{R}_l|\mathbf{x}_n}\Big)^2 + \frac{1}{2}\log\Big(\sigma^2_{\hat{R}_l|\mathbf{x}_n}\Big). \tag{2.23}$$

For the joint training of mean and variance, no additional ground-truth information is required. Instead, the network is able to increase the uncertainty which decreases the loss for uncertain samples during model training. This type of regression has recently been used in the context of object detection [40, 71, 16]. Therefore, it is possible to obtain semantic as well as spatial uncertainty within the scope of object detection.

2.3 Reasons for Unreliable Uncertainty

Recent work has found that modern neural networks tend to produce unreliable uncertainty information [11, 12, 13], i.e., the estimated uncertainty does not match the observed error distribution. The authors in [13] found that especially modern architectures with a high model capacity produce too overconfident confidence estimates, whereas simple models with a lower model capacity offer better calibration properties [13]. Furthermore, the authors study the effect of additional regularization techniques such as batch normalization [59] and weight decay [57] (for a description of these regularization techniques, cf. Sec. 2.1.1). The authors found that the use of batch normalization tends to increase miscalibration of neural networks [13] but the authors leave the interpretation of this phenomenon open for future work. Weight decay [57] is an additional penalty added to the loss term which penalizes large layer weights to prevent an overfitting of the network to the training data. As opposed to batch normalization, the authors in [13] found that weight decay tends to minimize the model miscalibration. We support this statement during our experiments as we observed that high network weights tend to produce less calibrated neural networks.

Finally, although the NLL is a direct measure for calibration [72, 73, 12, 74, 13] and used during model training, the authors in [13] observe that the NLL might disconnect from optimizing the calibration properties of the neural network during model training. This is in agreement with other works as they show that

the NLL can be decomposed into a classification and a refinement loss [72, 75, 74, 41]. As already pointed out in the last section, a neural network predicts a probability distribution targeting the desired outcome, e.g., a Bernoulli distribution within binary classification or a normal distribution within a regression setting. We further denote the neural network by $d(\mathbf{x})$ that predicts a probability distribution $f_{\hat{Y}}(\hat{y}|\mathbf{x})$ for a single sample with input $\mathbf{x} \in \mathcal{X}$ and the according ground-truth $\bar{y} \in \mathcal{Y}$, where $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \mathbb{R}$ within binary classification or regression, respectively. Furthermore, let $\hat{\boldsymbol{\vartheta}}_{\mathbf{x}}$ denote the estimated distribution parameters of $f_{\hat{Y}}(\hat{y}|\mathbf{x})$. For example, within binary classification, the network outcome is a Bernoulli distribution $f_{\hat{Y}}(\hat{y}|\mathbf{x}) = \text{Bern}(\hat{y}; \hat{p}_{\mathbf{x}})$ with confidence $\hat{p}_{\mathbf{x}}$, so that $\hat{\boldsymbol{\vartheta}}_{\mathbf{x}} = \{\hat{p}_{\mathbf{x}}\}$. In contrast, in the context of probabilistic regression the output is parameterized in terms of a normal distribution $f_{\hat{Y}}(\hat{y}|\mathbf{x}) = \mathcal{N}(\hat{y}; \mu_{\hat{y}|\mathbf{x}}, \sigma^2_{\hat{y}|\mathbf{x}})$ with mean $\mu_{\hat{y}|\mathbf{x}}$ and variance $\sigma^2_{\hat{y}|\mathbf{x}}$, so that $\hat{\boldsymbol{\vartheta}}_{\mathbf{x}} = \{\mu_{\hat{y}|\mathbf{x}}, \sigma^2_{\hat{y}|\mathbf{x}}\}$.

The predicted probability distribution is a direct estimate of the neural network about its uncertainty within the predicted outcome. Depending on the capacity of the used neural network, this estimation might not necessarily follow the real probability distribution about the ground-truth $f_{\bar{Y}}(\bar{y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})$ on the true data generation process given the network output $\hat{\boldsymbol{\vartheta}}_{\mathbf{x}}$. The network is trained by the NLL whose expectation over the data is denoted by $\mathbb{E}_{\mathbf{X},\bar{Y}\sim f_{\mathbf{X},\bar{Y}}}\Big[-\log\big(f_{\hat{Y}}(\bar{y}|\mathbf{x})\big)\Big]$[2]. In this sense, recent work has derived the decomposition of the expected NLL into calibration and refinement loss [72, 75, 74, 41] given by

$$\mathbb{E}_{\mathbf{X},\bar{Y}}[\text{NLL}] = \mathbb{E}_{\mathbf{X},\bar{Y}}\Big[-\log\big(f_{\hat{Y}}(\bar{y}|\mathbf{x})\big)\Big] \tag{2.24}$$

$$= \mathbb{E}_{\mathbf{X}}\Big[\mathbb{E}_{\bar{Y}}\Big[-\log\big(f_{\hat{Y}}(\bar{y}|\mathbf{x})\big) + \log\big(f_{\bar{Y}}(\bar{y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})\big) - \log\big(f_{\bar{Y}}(\bar{y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})\big)\Big]\Big] \tag{2.25}$$

$$= \mathbb{E}_{\mathbf{X}}\left[\mathbb{E}_{\bar{Y}}\left[\log\left(\frac{f_{\bar{Y}}(\bar{y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})}{f_{\hat{Y}}(\bar{y}|\mathbf{x})}\right)\right]\right] - \mathbb{E}_{\mathbf{X},\bar{Y}}\Big[-\log\big(f_{\bar{Y}}(\bar{y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})\big)\Big] \tag{2.26}$$

$$= \underbrace{\mathbb{E}_{\mathbf{X}}\Big[D_{\text{KL}}\big(f_{\bar{Y}}(\bar{Y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})||f_{\hat{Y}}(\bar{Y}|\mathbf{x})\big)\Big]}_{\text{Calibration loss}} + \underbrace{\mathbb{E}_{\mathbf{X},\bar{Y}}\Big[-\log\big(f_{\bar{Y}}(\bar{y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})\big)\Big]}_{\text{Refinement loss}}, \tag{2.27}$$

where

$$D_{\text{KL}}\big(f_{\bar{Y}}(\bar{Y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})||f_{\hat{Y}}(\bar{Y}|\mathbf{x})\big) = \int_{\mathcal{Y}} f_{\bar{Y}}(\bar{Y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}}) \log\left(\frac{f_{\bar{Y}}(\bar{Y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})}{f_{\hat{Y}}(\bar{y}|\mathbf{x})}\right) \mathrm{d}\bar{y} \tag{2.28}$$

is the Kullback-Leibler divergence between the uncertainty distribution $f_{\bar{Y}}(\bar{Y}|\hat{\boldsymbol{\vartheta}}_{\mathbf{x}})$ of the true data generation process given the network output $\hat{\boldsymbol{\vartheta}}_{\mathbf{x}}$ and the estimated uncertainty about the predicted outcome given by $f_{\hat{Y}}(\bar{Y}|\mathbf{x})$. The refinement loss is responsible to improve the network accuracy on the given ground-truth data set. The calibration loss serves as a regularizing term to match the predicted network confidence with the observed error. The authors in [13] assume that overfitting, which might be a direct cause of a model with too high capacity, might manifest in the calibration error rather than in the refinement loss during model training.

[2] For notational brevity, we further use an abbreviated expression to denote the random variable of the expectation, e.g., for $\mathbb{E}_{\mathbf{x}\sim f_{\mathbf{X}}}[\cdot]$, we write $\mathbb{E}_{\mathbf{x}}[\cdot]$.

The authors in [13] argue that once the model is not capable of further improving the network accuracy on the given data, the training process seeks to minimize the calibration loss which results in an overfitting to the training data.

Further related works [76, 77] seek to tackle this problem by directly incorporating additional regularization during model training. The authors in [76] introduce a regularization technique that is related to the Expected Calibration Error (ECE) known from classification calibration and use this technique to implement a more targeted regularization during model training. In addition, [77] study the effect of the focal loss [9] which is designed to set a higher weight on misclassified samples during model training. However, none of these approaches address the problem of overfitting during network training directly as they solely incorporate a penalty on high confidences but neglect the baseline generalization performance of a neural network. For example, in a binary classification setting, a neural network might perfectly predict the right output label for all samples in the training set. Since the regularization techniques proposed by [76] and [77] are based on the error on the training set, the probability mass is completely set to the correct label in this example and the additional regularization would not have any effect. This results in a perfect calibration on the training set. However, this behavior is commonly known as an overfitting on the training set. In this case, none of the regularization techniques is able to capture the "true" probability distribution on real-world data which is out of the training distribution. This overfitting behavior, which is also a lack of generalization, is a major reason for uncertainty miscalibration [13, 78].

A more recent study on the reasons for miscalibration by [78] examines the actual network architectures such as MLP-Mixer (fully-connected classification networks) [79], Transformer classification models [80], and modern ResNet [8] architectures. In their studies, the authors in [78] show that a higher model capacity does not necessarily lead to an increased miscalibration of models as modern architectures tend to improve the generalization performance [78]. Furthermore, their examinations are divided into in-distribution and out-of-distribution data for classification networks on several subsets of the ImageNet [81] data set. The authors in [78] found that a good calibration performance of a neural network also has a positive influence on the calibration on out-of-distribution data. Similar to [13], their examinations show a correlation between network capacity and miscalibration on in-distribution data which, however, is not as large as within the examinations of [13]. Interestingly, [78] also found that models with a higher generalization performance tend to have a better calibration performance on out-of-distribution data as these architectures seem to be more robust under domain shift [78].

3 Semantic Confidence Calibration

In the scope of image-based environment perception, we recently introduced the basic concepts of object detection models that are based on neural networks. A neural network estimates a score for each prediction that can be interpreted as a probability of correctness of the estimate. This confidence score reflects the aleatoric uncertainty about the semantic class membership for each prediction [82]. However, modern neural networks in the scope of classification are known to produce overconfident confidence estimates [12, 13], i.e., the predicted confidence does not match the observed accuracy. This is a major safety concern as such a bias within the confidence predictions might have a large impact on subsequent processes, e.g., object tracking. Thus, it is desirable to measure the misalignment between predicted confidence and observed accuracy. If we assert a systemic error within the objectness confidence, it is further possible to apply post-hoc calibration methods that seek to correct such a misalignment. For classification tasks, several methods such as Logistic Calibration [35], Temperature Scaling [13], or Beta Calibration [43] exist to correct biased confidence scores after model inference.

If we consider the more advanced tasks of object detection and segmentation, we face new challenges as we work with more complex models and within more complex environments. These tasks are of special interest as they are commonly used within the environment perception process, e.g., for safety-critical applications such as autonomous driving. For example, a detection model can be used within an object tracking process (cf. Chap. 6). In this context, the semantic confidence is used to decide which tracks are kept and which are discarded. Thus, a reliable self-assessment of the neural network about the semantic uncertainty is mandatory. However, if we detect a bias in the semantic confidence, calibration methods can be used to correct such a misalignment between predicted confidence and observed model performance. The concept of confidence calibration within an object tracking pipeline is schematically shown in Fig. 3.1. In this chapter, we solely focus on the task of object detection and examine the calibration properties of detection and segmentation

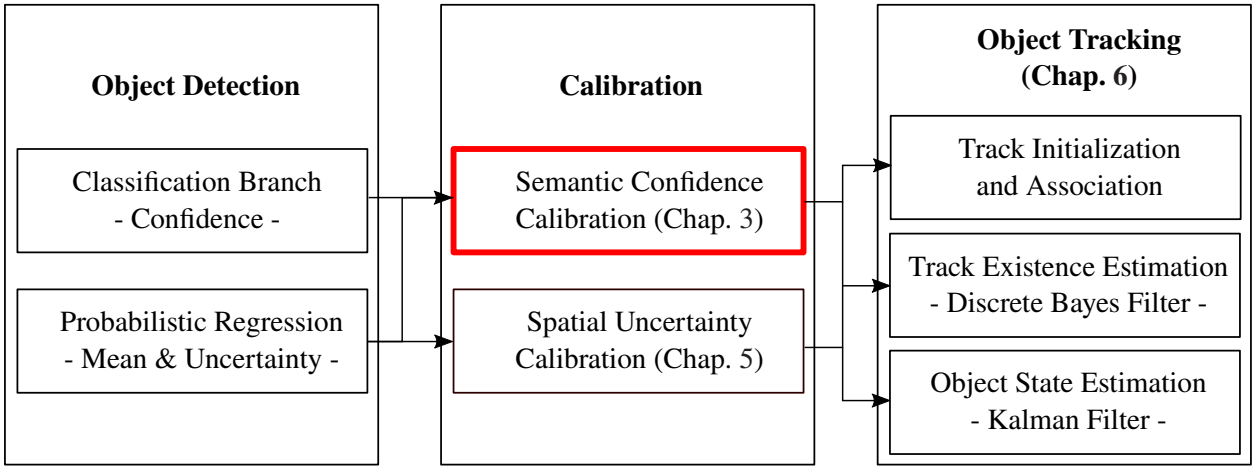


Figure 3.1: In this chapter, we focus on the evaluation and calibration of the semantic confidence that is of major importance for the management of tracks within a subsequent object tracking. Therefore, reliable and calibrated confidence information are of special interest.

models. Furthermore, we investigate novel calibration methods that are designed for detection and segmentation applications. These models are able to use additional information such as the object location or size for confidence recalibration. We investigate the influence of semantic confidence calibration to the task of object detection in Chap. 6.

We present the definition for classification calibration and transfer this to the task of object detection, instance segmentation, and semantic segmentation. These definitions are given in Sec. 3.2. Object detection is a joint task of classification and regression as a detection model does not only estimate the class membership but also the position of individual objects. Thus, we investigate the influence of the model's regression branch to the calibration properties and introduce an additional position-dependence to the definition for confidence calibration. Additionally, we extend common calibration metrics (Sec. 3.2) as well as common calibration methods (Sec. 3.3) to measure miscalibration and to apply post-hoc calibration for detection and segmentation models, respectively. This work has been subject of our previous publications in [1, 2, 4]. Finally, we evaluate our extended calibration methods as well as the definition for Bayesian confidence calibration in Sec. 3.4 for different neural network architectures and for different data sets. We summarize our contributions and findings in Sec. 3.5.

Contributions: We summarize our contributions in the following:

- Definition of confidence calibration for object detection, instance, and semantic segmentation.
- Multivariate extension of calibration methods to include additional position information.
- Extensive studies on the effect of position-dependent confidence calibration.

3.1 Related Work in the Context of Confidence Calibration

Recent works have started a discussion about the calibration of the class confidence of probabilistic forecasters [11, 72, 12, 13]. The common consent about the calibration of semantic uncertainty is that the estimated confidence of a classifier can be interpreted as a probability mass function (Bernoulli or categorical distribution for multivariate cases) which should reflect the model's uncertainty about the actual prediction. Thus, confidence calibration is defined as the task of matching the predicted confidence with the observed accuracy given a certain confidence level. It is possible to measure miscalibration using the expected calibration error (ECE) [12] which is derived from the definition of classification calibration. For ECE computation, all samples of a data set are grouped by their predicted confidence information into several equally sized distinct bins. Afterwards, it is possible to measure the accuracy within each bin. This yields multiple accuracy scores that are conditioned on a certain confidence range which is an approximation to the definition of calibration. Finally, an overall miscalibration score is computed using the weighted sum of the differences between observed accuracy and average confidence within each bin [12]. Besides the ECE, proper scoring rules such as Brier score or negative log likelihood also measure the calibration properties of a classifier inherently [83, 84]. Proper scoring rules are metrics that are minimized if and only if the predicted proba-

bility distribution is optimized towards the ground-truth data. These scoring rules can be decomposed into a calibration and a refinement part. We refer to [75, 83, 84] for a more detailed discussion.

Besides measuring miscalibration, recent work has developed methods for a subsequent (post-hoc) calibration of uncalibrated confidence estimates to calibrated ones. We distinguish between binning methods and scaling methods. Similar to the computation of the ECE, the binning methods group all samples by their predicted confidence and use a binning scheme to measure the accuracy conditioned on the confidence score. This binning scheme is then used to reassign the observed accuracy as the calibrated confidence to grouped samples. This technique is also referred to as Histogram Binning [36]. A more dynamic but related approach is Isotonic Regression [85] where a strictly monotonically increasing step function is fit to the training data. This function can be interpreted as a binning scheme with dynamic bin ranges and a dynamic amount of bins. Further extensions of these methods are Bayesian Binning into Quantiles (BBQ) [12] and Ensemble of Near Isotonic Regression (ENIR) [86] which are ensembles of multiple Histogram Binning and multiple Isotonic Regression models, respectively.

In contrast, scaling methods such as Logistic Calibration (or Platt scaling) [35], Temperature Scaling [13], Beta Calibration [43], or Dirichlet calibration [87] apply a rescaling of the logits before the sigmoid or softmax operation to yield calibrated confidence estimates. These scaling methods differ in their assumptions on the input data and the resulting recalibration scheme. Furthermore, Temperature Scaling [13] and Dirichlet calibration [87] are designed for the recalibration of multiclass classification tasks. The scaling methods are known to be sample efficient compared to the binning methods [88], i.e., these methods provide a qualitatively good calibration performance given only a small amount of samples, compared to the binning methods. However, these methods are also restricted by means of their underlying parametric assumptions [88], e.g., Gaussian distributions for Logistic Calibration [35, 43]. For this reason, the authors in [88] recently proposed a scaling-binning calibration method that combines the sample-efficiency of the scaling methods with the representational power of a Histogram Binning. A similar approach is applied by [89] who proposed a bin-wise Temperature Scaling for calibration. Besides the binning and scaling methods, the authors in [90] proposed a non-parametric Gaussian Process (GP) calibration scheme using a latent GP model with a categorical likelihood to perform recalibration for multiclass classification tasks. Another approach by [91] fits a spline function to the uncalibrated data to achieve confidence calibration.

Some research has also focused on confidence calibration directly during model training by regularization. The authors in [92] propose a confidence penalty during the training of a neural network model to reduce overconfidence. Similarly, the authors in [77] study the effect of the focal loss function [9] as the training objective to confidence calibration. They found that focal loss significantly reduces overconfidence but might also lead to underconfident models as well [2]. The drawback of these approaches is that they simply apply untargeted penalties to the confidence, regardless of the observed accuracy. This is addressed by [76] who propose the Maximum Mean Calibration Error (MMCE) which is a differentiable surrogate for the ECE and which can be used as a regularization term during model training. A common drawback of calibration methods during model training is that the baseline performance of a forecaster (e.g., accuracy or precision) is commonly way better on the training data compared to new samples during inference. This has an impact

on the regularization and commonly leads to a distortion of the confidences during model training as well. In this chapter, we focus on post-hoc calibration and especially on the standard calibration methods Histogram Binning [36], Logistic Calibration [35], and Beta Calibration [43], as these methods are most widely used. Furthermore, it is straightforward to utilize and extend these methods to our task of object detection and segmentation calibration.

Recently, the authors in [93] proposed the concept of meta classification which is quite related to our approach of extended and feature-aware confidence calibration. Within meta classification, different sources of uncertainty are aggregated which are provided by a forecaster. Afterwards, a simple classification model (e.g., Logistic Calibration) is used to produce the final outcome. This concept has been extended to object detection by [38] which is denoted as MetaDetect and utilizes further features such as bounding box position or object size. In the context of this work, we focus on the characteristics of calibration. Furthermore, we derive our extended methods directly from existing calibration methods and set them in an overall context. In addition, we present a multivariate extension of the calibration methods that also allows for the detection of possible correlations between the individual features.

3.2 Definitions and Metrics for Confidence Calibration

In this section, we present the definition of semantic confidence calibration for classification and extend it to the tasks of object detection, instance segmentation, and semantic segmentation. The derivation of these definitions was the subject of our research in [1], [4, p. 228 f.], and [4, p. 230 f.].

3.2.1 Classification

Let $\mathcal{D}$ denote a data set of size N which consists of several images $\mathbf{X} \in \mathcal{X}$ with height H, width W, and number of channels C. Each image belongs to a certain ground-truth class $\bar{Y} \in \mathcal{Y} = \{1, ..., K\}$, so that the joint distribution for the input $\mathbf{X}$ and the respective ground-truth label $\bar{Y}$ is given by $f_{\mathbf{X},\bar{Y}}(\mathbf{x}, \bar{y}) = f_{\bar{Y}}(\bar{y}|\mathbf{x}) f_{\mathbf{X}}(\mathbf{x})$. A multiclass classification model seeks to approximate $f_{\bar{Y}}(\bar{y}|\mathbf{x})$ by learning its parameters so that it is able to predict a label $\hat{Y} \in \mathcal{Y}$ with a certain confidence score $\hat{P} \in [0, 1] = [0, 1]$ that represents the model's belief about the prediction's correctness. These predictions follow the joint model distribution $f_{\hat{P},\hat{Y}}(\hat{p}, \hat{y}|\mathbf{x})$. A classification model is confidence calibrated, if

$$\mathbb{P}(\hat{Y} = \bar{Y}|\hat{P} = \hat{p}) = \hat{p} \tag{3.1}$$

is fullfilled for all $\hat{p} \in [0, 1]$ [11, 12, 13]. This definition implies that the observed accuracy $\mathbb{P}(\hat{Y} = \bar{Y}|\hat{P} = \hat{p})$ should match the estimated confidence given a certain confidence level $\hat{p}$. For example, given 100 predictions with a confidence score of 0.8 each, we would expect an accuracy of also 80%. If we observe a deviation, a model is said to be miscalibrated. Note that for binary classification with $\mathcal{Y} = \{0, 1\}$, the network output of a classification model is commonly a sigmoidal function that outputs a score in the $(0, 1)$ interval, indicating

the belief for $\hat{Y} = 1$. Thus, our calibration target is the observed *relative frequency* for $\bar{Y} = 1$ instead of the accuracy, so that

$$\mathbb{P}(\bar{Y} = 1 | \hat{P} = \hat{p}) = \hat{p} \tag{3.2}$$

is required for all $\hat{p} \in [0, 1]$.

As already stated in the introduction in Sec. 2.2.1, the predicted probability distribution for $\hat{Y}$ can be expressed in terms of a Bernoulli distribution with confidence $\hat{P}$ as the probability parameter, so that the estimated probability mass function is defined by

$$f_{\hat{Y}}(\hat{y}|\hat{p}) = \text{Bern}(\hat{y}; \hat{p}) = \hat{p}^{\hat{y}}(1 - \hat{p})^{1-\hat{y}}. \tag{3.3}$$

The definition for confidence calibration in (3.1) can be used to derive the Expected Calibration Error (ECE) [12] that is a common metric to measure miscalibration in the scope of classification [12, 13]. We seek to minimize the expectation of the difference between predicted confidence and observed accuracy [12] which is denoted by

$$\mathbb{E}_{\hat{P} \sim f_{\hat{P}}} \Big[\big| \mathbb{P}(\hat{Y} = \bar{Y} | \hat{P} = \hat{p}) - \hat{p} \big| \Big] \tag{3.4}$$

$$= \int_0^1 f_{\hat{R}}(\hat{p}) \cdot \big| \mathbb{P}(\hat{Y} = \bar{Y} | \hat{P} = \hat{p}) - \hat{p} \big| \mathrm{d}\hat{p} \tag{3.5}$$

$$= \int_0^1 \big| \mathbb{P}(\hat{Y} = \bar{Y} | \hat{P} = \hat{p}) - \hat{p} \big| \mathrm{d}F_{\hat{P}}(\hat{p}), \tag{3.6}$$

using the Rieman-Stieltjes integral and with $F_{\hat{P}}(\hat{p})$ as the Cumulative Density Function (CDF) of the estimated confidence distribution. However, since $\hat{P}$ is a continuous random variable, we can not get the probability in (3.4) using a finite set of samples [13]. Therefore, the ECE is approximated by a binning scheme over estimated confidence distribution of $\hat{P} \in [0, 1]$ using I equally sized bins. Given a finite data set, the ECE [12], [13, p. 11] is an approximation of (3.6) given by

$$\text{ECE} := \sum_{i=1}^{I} \mathbb{P}(\hat{P} \in B_i) \cdot \big| \mathbb{P}(\hat{Y} = \bar{Y} | \hat{P} = \hat{p}_i) - \hat{p}_i \big|, \tag{3.7}$$

$$\forall \hat{p}_i \in B_i, i \in \{1, \dots I\}.$$

where $\hat{p}_i$ denote all possible confidences within interval B_i. In other words, the ECE is calculated using the weighted sum of the differences between average confidence and observed accuracy/frequency over all bins

using a finite set of samples. This yields the representation form

$$\mathrm{ECE} = \sum_{i=1}^{I} \frac{N_i}{N} |\mathrm{acc}(i) - \mathrm{conf}(i)|, \tag{3.8}$$

where N_i denotes the number of samples within bin B_i and

$$\mathrm{acc}(i) = \frac{1}{N_i} \sum_{n_i=1}^{N_i} \mathbb{1}(\hat{y}_{n_i} = \bar{y}_{n_i}), \tag{3.9}$$

$$\mathrm{conf}(i) = \frac{1}{N_i} \sum_{n_i=1}^{N_i} \hat{p}_{n_i}, \tag{3.10}$$

with indicator function $\mathbb{1}(\cdot)$ [12, 13]. In this case, $\mathrm{acc}(i)$ and $\mathrm{conf}(i)$ denote the average accuracy (or frequency within binary classification) and the average confidence within each bin, respectively.

3.2.2 Object Detection

Similar to a classifier, an object detection model, which is based on a neural network, also estimates a label $\hat{Y} \in \mathcal{Y}$ and an according objectness confidence score $\hat{P} \in [0, 1]$ for each prediction within a single image $\mathbf{X} \in \mathcal{X}$. Let further denote $\bar{\mathbf{R}} \in \mathcal{R} = [0, 1]^L$ the ground-truth information for the object's position (relative to image size) where L denotes the dimension of the box encoding (commonly center x and y positions c_x, c_y as well as width w and height h). Thus, the joint ground-truth data distribution extends to $f_{\mathbf{X},\bar{Y},\bar{\mathbf{R}}}(\mathbf{x}, \bar{y}, \bar{\mathbf{r}}) = f_{\bar{Y},\bar{\mathbf{R}}}(\bar{y}, \bar{\mathbf{r}}|\mathbf{x}) f_{\mathbf{X}}(\mathbf{x})$. An object detection model thus also needs to infer the object's position to approximate $f_{\bar{Y},\bar{\mathbf{R}}}(\bar{y}, \bar{\mathbf{r}}|\mathbf{x})$. In the following, we denote the position predictions as $\hat{\mathbf{R}} \in \mathcal{R}$, so that the overall output distribution of an object detection model is denoted by $f_{\hat{P},\hat{Y},\hat{\mathbf{R}}}(\hat{p}, \hat{y}, \hat{\mathbf{r}}|\mathbf{x})$. In contrast to the simple case of classification calibration, there are some limitations we need to address within object detection calibration. On the one hand, as we know from Sec. 2.1, most object detectors use anchor or proposal boxes to implement the object detection. These boxes serve as an initial estimate for a possible object location and get scaled and shifted in a subsequent step to match a real object. However, if only a few amount of objects is present within an image, most anchor/proposal boxes are classified as background. These predictions are discarded in the final post-processing of a common object detection pipeline. This approach significantly reduces the model output to the user-relevant predictions. During the model evaluation, we can denote the amount of missed objects, the so-called *false negatives*. Since we have no information about what the detection model has classified as background, we can not denote the *true negatives* within a single frame. However, this information is mandatory for accuracy computation. For this reason, we further use the precision as our calibration target as it measures the fraction of correctly identified objects. The precision denotes the fraction of correctly detected and classified objects given all predicted objects by the detection model. In order to determine the precision, an assignment of the predictions to ground truth objects is necessary. This is usually done using the Intersection over Union (IoU) score between the predicted and the ground-truth

objects. Furthermore, this means that the precision and thus the calibration of the detection model depends on the selected IoU score.

On the other hand, as we observed in our examinations in [1] and [4], the data distribution of the training set during model training has an influence on the prediction performance of an object detector. For example, if most of the training samples have been located within the image center, the predictions near the boundaries might have a reduced performance. Therefore, we introduce a position-dependency to the definition of confidence calibration for object detection which is thus given by

$$\mathbb{P}(\hat{M} = 1 | \hat{P} = \hat{p}, \hat{Y} = \hat{y}, \hat{\mathbf{R}} = \hat{\mathbf{r}}) = \hat{p} \tag{3.11}$$

$$\forall \hat{p} \in [0, 1], \hat{y} \in \mathcal{Y}, \hat{\mathbf{r}} \in \mathcal{R}.$$

In this case, $\hat{M} \in \{0, 1\}$ denotes if a predicted object has matched a ground-truth object with a certain IoU and thus $\mathbb{P}(\hat{M} = 1)$ is a shorthand notation for $\mathbb{P}(\hat{Y} = \bar{Y}, \hat{\mathbf{R}} = \bar{\mathbf{R}})$.

Similar to the standard ECE definition in (3.4), we derive the Detection Expected Calibration Error (D-ECE) which is the extension of the ECE to object detection. Let $\hat{\mathbf{s}} = (\hat{p}, \hat{y}, \hat{\mathbf{r}})$ denote a single prediction so that $\hat{\mathbf{s}} \in \mathcal{S}$, where $\mathcal{S}$ is the aggregated set of the confidence, label, and bounding box spaces. If an object detection model is calibrated, the expected difference between predicted confidence and observed precision conditioned on the model output gets minimal. The predicted joint output distribution is denoted by $f_{\hat{P},\hat{Y},\hat{\mathbf{R}}}(\hat{p}, \hat{y}, \hat{\mathbf{r}}) = f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}})$, where $F_{\hat{P},\hat{Y},\hat{\mathbf{R}}}(\hat{\mathbf{s}})$ is the respective CDF function. Thus, the expectation is denoted by

$$\mathbb{E}_{\hat{P},\hat{Y},\hat{\mathbf{R}}}\Big[\big|\mathbb{P}(\hat{M} = 1 | \hat{P} = \hat{p}, \hat{Y} = \hat{y}, \hat{\mathbf{R}} = \hat{\mathbf{r}}) - \hat{p}\big|\Big] \tag{3.12}$$

$$= \mathbb{E}_{\hat{\mathbf{S}}}\Big[\big|\mathbb{P}(\hat{M} = 1 | \hat{\mathbf{S}} = \hat{\mathbf{s}}) - \hat{p}\big|\Big] \tag{3.13}$$

$$= \int_{\mathcal{S}} f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}) \cdot \big|\mathbb{P}(\hat{M} = 1 | \hat{\mathbf{S}} = \hat{\mathbf{s}}) - \hat{p}\big| \mathrm{d}\hat{\mathbf{s}} \tag{3.14}$$

$$= \int_{\mathcal{S}} \big|\mathbb{P}(\hat{M} = 1 | \hat{\mathbf{S}} = \hat{\mathbf{s}}) - \hat{p}\big| \mathrm{d}F_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}). \tag{3.15}$$

Similar to (3.7), we can approximate the integral by a multidimensional binning scheme with I distinct bins $\mathbf{B}_i$ over the joint output space $\mathcal{S}$, so that the D-ECE is an approximation of the integral in (3.15) by

$$\text{D-ECE} := \sum_{i=1}^{I} \mathbb{P}(\hat{\mathbf{S}} \in \mathbf{B}_i) \cdot \big|\mathbb{P}(\hat{M} = 1 | \hat{P} = \hat{p}_i, \hat{Y} = \hat{y}_i, \hat{\mathbf{R}} = \hat{\mathbf{r}}_i) - \hat{p}_i\big| \tag{3.16}$$

$$\forall \hat{p}_i, \hat{y}_i, \hat{\mathbf{r}}_i \in \mathbf{B}_i, \quad i \in \{1, \dots I\},$$

which is finally computed by the weighted sum

$$\text{D-ECE} = \sum_{i=1}^{I} \frac{N_i}{N} |\text{prec}(i) - \text{conf}(i)|, \tag{3.17}$$

with N_i as the number of samples within bin $\mathbf{B}_i$ and $\text{prec}(i)$ as the precision within bin $\mathbf{B}_i$. Thus, the D-ECE can be seen as the weighted sum of differences between precision and average confidence over all bins in the output space for a finite set of samples.

3.2.3 Instance Segmentation

Let $\mathcal{V}$ denote the set of all objects over all images in data set $\mathcal{D}$, where V denotes the total amount of objects. Instance segmentation is the combined task of predicting individual objects and their shape at the pixel-level. Thus, for each pixel $j \in \mathcal{J}_v = \{1, ..., J_v\}$ within the predicted bounding box $\hat{\mathbf{R}}_v \in \mathcal{R}$ of an estimated object $v \in \mathcal{V}$ with predicted label $\hat{Y}_v \in \mathcal{Y}$, an instance segmentation model also predicts a label $\hat{Y}_j^* \in \mathcal{Y}^* = \{0, 1\}$ in conjunction with a confidence score $\hat{P}_j^* \in [0, 1]$, indicating the estimated membership of each pixel to the object mask. We further denote $\bar{Y}_j^*$ as the ground-truth information for the object segmentation masks and $\mathbf{R}_j^* \in \mathcal{R}^* = [0, 1]^{L^*}$ as the (normalized) pixel-position within a bounding box $\hat{\mathbf{R}}_v$, where L^* denotes the size of the used position-encoding for a single pixel. In contrast to object detection, it is possible to compute the accuracy on pixel-level over all objects and all instance segmentation masks. Since the mask inference reduces to a binary classification task, our calibration target for instance segmentation is the pixel-wise relative frequency $\mathbb{P}(\hat{Y}_j^* = 1)$ of each pixel belonging to the object's shape. Similar to the definition of object detection calibration, we further introduce a position-dependency. For example, pixels close to the shape boundary might have different calibration properties. Therefore, confidence calibration for instance segmentation is defined by

$$\begin{aligned} &\mathbb{P}(\hat{Y}_j^* = 1 | \hat{P}_j^* = \hat{p}^*, \mathbf{R}_j^* = \hat{\mathbf{r}}^*, \hat{Y} = \hat{y}) = \hat{p}^*, \\ &\forall \hat{p}^* \in [0, 1], \hat{\mathbf{r}}^* \in \mathcal{R}^*, \hat{y} \in \mathcal{Y}, j \in \mathcal{J}_v, v \in \mathcal{V}. \end{aligned} \tag{3.18}$$

Similar to (3.16), we can use the D-ECE to measure the miscalibration of an instance segmentation model. For this purpose, a binning scheme over the joint space for the pixel-wise confidence, the pixel position $\mathcal{R}^*$, and all possible labels is used to approximate the D-ECE.

3.2.4 Semantic Segmentation

In contrast to instance segmentation, a semantic segmentation model does not predict individual objects but rather estimates a label $\hat{Y}_j^* \in \mathcal{Y} = \{1, ..., K\}$ with corresponding confidence $\hat{P}_j^* \in [0, 1]$ for each pixel $j \in \mathcal{J}$ in an input image $\mathbf{X} \in \mathcal{X}$, where $\bar{Y}_j^* \in \mathcal{Y}$ denotes the ground-truth information for pixel j. Therefore, semantic segmentation can be seen as a joint (multiclass) classification task for each pixel in an image. Thus,

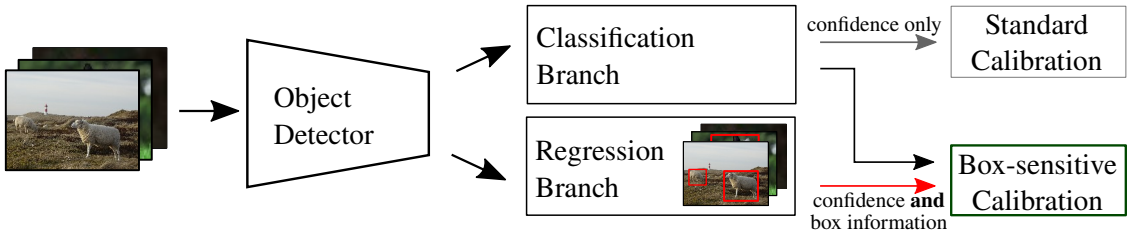


Figure 3.2: Concept of box-sensitive confidence calibration w.r.t. the regression branch of an object detector [1, p. 2, Fig. 2]. A detection model predicts several objects within an image with a certain position and a certain class. Standard calibration methods (top row) only use the classification branch of a detection model for confidence calibration. In contrast, our position-dependent calibration methods also utilize the additional regression branch to keep track of possible correlations between box position and miscalibration.

the definition of confidence calibration for semantic segmentation changes to

$$\mathbb{P}(\hat{Y}_j^* = \bar{Y}_j^* | \hat{P}_j^* = \hat{p}^*, \mathbf{R}_j^* = \hat{\mathbf{r}}^*) = \hat{p}^*, \tag{3.19}$$

$$\forall \hat{p}^* \in [0,1], \hat{\mathbf{r}}^* \in \mathcal{R}^*, j \in \mathcal{J}.$$

In contrast to the definition for instance segmentation calibration in (3.18), we have no dependency on individual objects. Thus, we the definition of calibration for semantic segmentation is related to the calibration definition for classification in (3.1) but on pixel-level and with an additional dependency on the pixel position.

3.3 Multivariate Confidence Calibration

The task of confidence calibration is to remap probability estimates so that they reflect the observed accuracy, frequency, or precision. These calibration methods are applied as post-hoc calibration that is applied after model inference. In a first step, we review existing calibration techniques and group these methods into binning and scaling methods. Scaling calibration methods perform post-hoc recalibration by rescaling the output of a forecaster to achieve well-calibrated confidence estimates [35, 13]. In contrast, binning methods divide the probability space into several distinct bins (similar to the calculation of the ECE) and use this binning scheme to derive a calibrated confidence [36, 12].

Moreover, we refer to the definition for position-dependent calibration from the previous section and extend these methods to perform confidence calibration by means of the additional position information which is provided by an object detection or segmentation model. This concept is schematically shown in Fig. 3.2. The advantage of these extended calibration methods is that they are sensitive to possible correlations between miscalibration and object location/shape. If there is a position dependence in the confidence miscalibration, the common calibration methods will not be able to perform a proper recalibration. The effect of position-dependent calibration is demonstrated in an artificial example in Fig. 3.3.

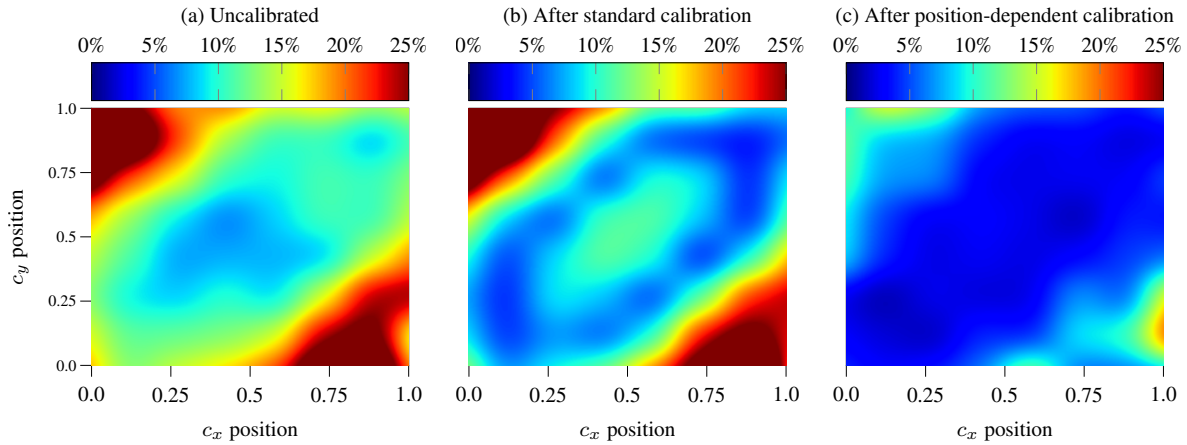


Figure 3.3: Qualitative example of multivariate confidence calibration on an artificially created data set [1, p. 5, Fig. 3], [4, p. 232, Fig. 1]. The samples are drawn with a certain confidence score $\hat{P}$ and a matched flag $\hat{M} \in \{0, 1\}$. Similar to the computation of the D-ECE, the position space is divided into several bins. We measure the gap between average confidence and observed precision within each bin and visualize the position-dependent miscalibration in terms of a heatmap. (a) The deviation between average confidence and average precision follows a correlated bivariate normal distribution and increases towards the distribution boundary. (b) Common calibration methods are only able to rescale to average confidence in this scenario. This yields new confidences which are calibrated on average but still show a position-dependent calibration error. (c) In contrast, our multivariate calibration methods are able to successfully recalibrate the confidence information by means of the position information.

The extended calibration methods yield a calibrated confidence $\hat{Q} \in [0, 1]$ which reflects the probability of a predicted object to match an existing one given the uncalibrated confidence $\hat{P}$, the object category $\hat{Y}$, and the position information $\hat{\mathbf{R}}$ in the joint space $\mathcal{S}$, so that $h : \mathcal{S} \to [0, 1]$. Therefore, the predicted probability distribution for the categorical random variable $\hat{M}$ (object detection) after calibration is also a Bernoulli distribution whose probability parameter $\hat{Q}$ can be expressed as a function of the uncalibrated confidence and the predicted position information, so that the Probability Mass Function (PMF) of $\hat{M}$ is given by

$$\mathbb{P}\big(\hat{m}|h(\hat{p}, \hat{y}, \hat{\mathbf{r}})\big) = \text{Bern}\big(\hat{m}; h(\hat{p}, \hat{y}, \hat{\mathbf{r}})\big). \tag{3.20}$$

In the context of the multivariate scaling methods, we further distinguish between conditionally independent and dependent variants. Both variants are able to model the influence of additional position information to the calibration output. However, the conditionally independent methods assume independent random variables as input which simplifies the calibration computation but also leads to a reduced representational power. In contrast, the conditional dependent methods model the input distribution as a joint multivariate probability distribution so that it is possible to capture possible correlations between the input quantities.

The multivariate extensions of the calibration methods was subject of our work in [1]. In the following, we derive the position-dependent calibration techniques for the binning and scaling methods, respectively.

3.3.1 Histogram Binning

Similar to the approximate computation of the ECE (cf. Sec. 3.2), the Histogram Binning method uses a binning scheme with I equally sized bins over the confidence space $[0, 1]$ to group all samples in $\mathcal{D}$ by their confidence [36]. Afterwards, it is possible to measure the accuracy, frequency, or precision w.r.t. the confidence. This allows for a remapping of uncalibrated confidences to calibrated ones [36]. More formally, let N denote the amount of predictions obtained by a neural network with a certain label and confidence. Additionally, let I denote the number of bins with interval boundaries $0 = a_1 < a_2 < ... < a_{I+1} = 1$ as well as the according calibration parameters $\boldsymbol{\theta} = \{\theta_i | i \in \{1, ..., I\}\}$, which reflect the observed accuracy, frequency, or precision within each bin. The objective of Histogram Binning is the minimization of

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{n=1}^{N} \sum_{i=1}^{I} \mathbb{1}(\hat{p}_n \in B_i)(\theta_i - \bar{y}_n)^2, \tag{3.21}$$

to infer the optimal recalibration parameters $\hat{\boldsymbol{\theta}}$, where $\mathbb{1}(\hat{p}_n \in B_i)$ is the indicator function that a predicted sample falls into bin B_i [36]. By the strong law of large numbers, each parameter θ_i converges to the fraction of positive samples within each bin [36, 13]. In contrast to isotonic regression [85], Histogram Binning does not guarantee a monotonically increasing mapping from uncalibrated to calibrated confidence scores as there is no restriction for neighboring bins to yield smaller or larger recalibration parameters θ_i. Therefore, Histogram Binning might also affect the order of the predicted samples and thus might have an influence on the computation of the average precision score.

Similar to the multivariate extension of the ECE in (3.16), we can further use a multivariate binning scheme over the joint space $\mathcal{S}$ which consists of the probability space $[0, 1]$, all possible labels $\mathcal{Y}$, and the spatial space $\mathcal{R}$. In this way, all samples in $\mathcal{D}$ are grouped in I distinct bins $\mathbf{B}_i$ by their confidence, label, and position information to construct a recalibration mapping, so that the objective function slightly changes to

$$\hat{\boldsymbol{\theta}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \sum_{n=1}^{N} \sum_{i=1}^{I} \mathbb{1}(\hat{\mathbf{s}}_n \in \mathbf{B}_i)(\theta_i - \bar{y}_n)^2, \tag{3.22}$$

where $\mathbb{1}(\hat{\mathbf{s}}_n \in \mathbf{B}_i)$ is the indicator function that sample n falls into bin $\mathbf{B}_i$ with a certain label in $\mathcal{Y}$ and a certain set of boundaries for the confidence in $[0, 1]$ and the position information in $\mathcal{R}$.

3.3.2 Scaling Methods

In contrast to binning methods such as Histogram Binning, scaling methods perform a rescaling of the network output to yield calibrated confidences. For this purpose, the methods Logistic Calibration [35] and Beta Calibration [43] either rescale the logit before applying a sigmoid (or softmax for multiclass problems) or directly rescale the confidence output after the sigmoid or softmax function, respectively. We further denote the logit by $Z \in \mathbb{R}$ which is related to the same distribution as the predicted confidence $\hat{P}$, so that the predicted joint model distribution is given by $f_{Z,\hat{Y},\hat{\mathbf{R}}}(z, \hat{y}, \hat{\mathbf{r}})$.

The advantage of scaling methods is that they require significantly less parameters compared to binning methods and thus are able to achieve a good calibration results given less data. We further inspect the rescaling of the confidence output in more detail. For binary classification, the distribution of the confidence $\hat{P} \in [0, 1]$ can be represented using the probability density functions $f_{\hat{P}}(\hat{p}|\bar{Y} = 1)$ and $f_{\hat{P}}(\hat{p}|\bar{Y} = 0)$ for the positive "+" ($\bar{Y} = 1$) and the negative classes "−" ($\bar{Y} = 0$), respectively. If we further treat these density functions as the likelihood for $\bar{Y}$ given $\hat{P}$, we can use this representation to derive a recalibration scheme by the likelihood ratio between the likelihood for $\bar{Y} = 1$ and $\bar{Y} = 0$ [43]. Using the logarithm of this ratio, we further refer to the log likelihood ratio as

$$\ell r(\hat{p}) = \log \frac{f_{\hat{P}}(\hat{p}|\bar{Y} = 1)}{f_{\hat{P}}(\hat{p}|\bar{Y} = 0)}. \tag{3.23}$$

The calibrated probability for $\mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p})$ can be derived by the ratio

$$\frac{\mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p})}{\mathbb{P}(\bar{Y} = 0|\hat{P} = \hat{p})} = \frac{f_{\hat{P}}(\hat{p}|\bar{Y} = 1)}{f_{\hat{P}}(\hat{p}|\bar{Y} = 0)} \frac{\mathbb{P}(\bar{Y} = 1)}{\mathbb{P}(\bar{Y} = 0)}. \tag{3.24}$$

If we assume a uniform prior for the positive and negative classes so that $\mathbb{P}(\bar{Y} = 1) = \mathbb{P}(\bar{Y} = 0)$, the ratio $\frac{\mathbb{P}(\bar{Y}=1)}{\mathbb{P}(\bar{Y}=0)}$ evaluates to 1 and can be neglected [43]. In addition, if $\mathbb{P}(\bar{Y} = 0|\hat{P} = \hat{p}) = 1 - \mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p})$ within binary classification, the likelihood ratio equals the posterior odds [43], [94, p. 279], and we can derive a calibrated probability by

$$\frac{\mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p})}{\mathbb{P}(\bar{Y} = 0|\hat{P} = \hat{p})} = \frac{f_{\hat{P}}(\hat{p}|\bar{Y} = 1)}{f_{\hat{P}}(\hat{p}|\bar{Y} = 0)} \tag{3.25}$$

$$\Leftrightarrow \quad \mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p}) = \big(1 - \mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p})\big) \exp\big(\ell r(\hat{p})\big) \tag{3.26}$$

$$\Leftrightarrow \quad \mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p}) = \exp\big(\ell r(\hat{p})\big) - \mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p}) \exp\big(\ell r(\hat{p})\big) \tag{3.27}$$

$$\Leftrightarrow \quad \frac{\exp\big(\ell r(\hat{p})\big)}{\mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p})} = 1 + \exp\big(\ell r(\hat{p})\big) \tag{3.28}$$

$$\Leftrightarrow \quad \frac{\mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p})}{\exp\big(\ell r(\hat{p})\big)} = \frac{1}{1 + \exp\big(\ell r(\hat{p})\big)} \tag{3.29}$$

$$\Leftrightarrow \quad \mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p}) = \frac{\exp\big(\ell r(\hat{p})\big)}{1 + \exp\big(\ell r(\hat{p})\big)} = \Phi\big(\ell r(\hat{p})\big), \tag{3.30}$$

which recovers the logistic (sigmoid) function $\Phi\big(\ell r(\hat{p})\big)$ [43]. This derivation also holds if we consider the logits Z instead of the confidence $\hat{P}$, i.e., for the derivation of the Logistic Calibration function. In this case, we utilize the log likelihood ratio for the logits by

$$\ell r(z) = \log \frac{f_Z(z|\bar{Y} = 1)}{f_Z(z|\bar{Y} = 0)}, \tag{3.31}$$

which actually yields the probability for $\mathbb{P}(\bar{Y} = 1|Z = z)$. However, common literature treats this formulation equivalently to $\mathbb{P}(\bar{Y} = 1|\hat{P} = \hat{p})$ since we are interested in the calibration properties by means of an interpretable confidence score. This equivalent interpretation for calibration holds as the logit Z is commonly remapped to a confidence $\hat{P} = \Phi(Z)$ using the bijective sigmoid function. Since the recalibration mapping is also a logistic and thus monotonically increasing function, the standard scaling methods do not affect the order of the samples provided by a forecaster. Therefore, the calibration does not affect the computation of the average precision score.

According to the definition of confidence calibration for object detection models in Sec. 3.2, we further want to include additional information such as position and shape of the predicted objects into a calibration mapping for detection models. For this reason, we can also use the derivation of a calibrated probability in (3.30) for multivariate probability distributions given the joint distribution $f_{\hat{P},\hat{Y},\hat{\mathbf{R}}}(\hat{p}, \hat{y}, \hat{\mathbf{r}}|\hat{M})$. We further use the shorthand notation $f_{\hat{\mathbf{S}}_{\hat{P}}}(\hat{\mathbf{s}}_{\hat{P}}|\hat{M})$ for $\hat{\mathbf{S}}_{\hat{P}} = (\hat{P}, \hat{Y}, \hat{\mathbf{R}}) \in \mathcal{S}_{\hat{P}}$. The equivalent joint distribution for the logits is further denoted by $f_{\hat{\mathbf{S}}_Z}(\hat{\mathbf{s}}_Z|\hat{M})$ for $\hat{\mathbf{S}}_Z = (Z, \hat{Y}, \hat{\mathbf{R}}) \in \mathcal{S}_Z$. To derive a multivariate calibration mapping with K dimensions, we can assume conditional independence between all quantities in $\mathcal{S}_{\hat{P}}$ or $\mathcal{S}_Z$ so that the log likelihood ratio can be rewritten to

$$\ell r(\hat{\mathbf{s}}) = \sum_{k=1}^{K} \log \frac{f_{\hat{\mathbf{S}}}(\hat{s}_k|\hat{M} = 1)}{f_{\hat{\mathbf{S}}}(\hat{s}_k|\hat{M} = 0)}, \tag{3.32}$$

for any $\hat{\mathbf{S}}$ either in $\mathcal{S}_{\hat{P}}$ or $\mathcal{S}_Z$. In contrast, it is also possible to derive a calibration mapping assuming dependencies between all quantities in $\mathcal{S}_{\hat{P}}$ or $\mathcal{S}_Z$ using the log likelihood ratio for multivariate distributions. In the following, we will derive the multivariate extension for the calibration methods Logistic Calibration [35] and Beta Calibration [43].

Logistic Calibration

A popular recalibration method for binary classification is Logistic Calibration (or Platt scaling) [35] which assumes normally distributed logit scores with mean values $\mu^+, \mu^- \in \mathbb{R}$ for the positive and negative classes, respectively, and equal variance $\sigma^2 \in \mathbb{R}_{>0}$. Therefore, the probability density functions are defined as $f_Z(z|\bar{Y} = 1) = \mathcal{N}(z; \mu^+, \sigma^2)$ and $f_Z(z|\bar{Y} = 0) = \mathcal{N}(z; \mu^-, \sigma^2)$ [35, 43]. The log likelihood ratio $\ell r(z)$ is thus given by

$$\ell r(z) = \log \frac{f_Z(z|\bar{Y} = 1)}{f_Z(z|\bar{Y} = 0)} = \frac{1}{2\sigma^2}\Big[(z - \mu^-)^2 - (z - \mu^+)^2\Big] \tag{3.33}$$

$$= \frac{1}{2\sigma^2}\Big[2z(\mu^+ - \mu^-) - (\mu_+^2 - \mu_-^2)\Big] \tag{3.34}$$

$$= \frac{\mu^+ - \mu^-}{2\sigma^2}\Big[z - (\mu^+ - \mu^-)\Big] \tag{3.35}$$

$$= w(z - \eta), \tag{3.36}$$

where $w = \frac{1}{2\sigma^2}(\mu^+ - \mu^-)$ and $\eta = \mu^+ - \mu^-$ [43]. In practice, Logistic Calibration utilizes a disentangled representation by the scale weight $w \in \mathbb{R}_{>0}$ and a bias $\delta \in \mathbb{R}$ which can be interpreted as $\delta = -\eta w$. These parameters are obtained by Maximum Likelihood Estimation (MLE) using the Negative Log Likelihood (NLL) loss and are used to rescale the logits Z, so that a calibrated confidence estimate is derived by

$$\mathbb{P}\big(\bar{Y} = 1 | \hat{P} = \Phi(z)\big) = \Phi(w \cdot z + \delta), \tag{3.37}$$

[35]. Similarly, we can derive the multivariate extension of the log likelihood ratio assuming conditional independence between all quantities, so that the likelihood ratio is derived in the sense of (3.32) using normal distributions for each quantity k which yields the log likelihood ratio

$$\ell r(\hat{\mathbf{s}}_Z) = \hat{\mathbf{s}}_Z^\top \mathbf{w} + \delta, \tag{3.38}$$

where $\mathbf{w} \in \mathbb{R}^K$. In contrast, if we assume conditional dependence between all quantities, the log likelihood ratio $\ell r(\hat{\mathbf{s}})$ is represented as the fraction of two Gaussians with mean vectors $\boldsymbol{\mu}^+, \boldsymbol{\mu}^- \in \mathbb{R}^K$ for the positive and negative classes, respectively, and the positive semidefinit covariance matrices $\boldsymbol{\Sigma}^+, \boldsymbol{\Sigma}^- \in \mathbb{R}^{K \times K}$. This yields the log likelihood ratio

$$\ell r(\hat{\mathbf{s}}_Z) = \frac{1}{2}\Big[(\hat{\mathbf{s}}_-^\top \boldsymbol{\Sigma}_-^{-1} \hat{\mathbf{s}}^-) - (\hat{\mathbf{s}}_+^\top \boldsymbol{\Sigma}_+^{-1} \hat{\mathbf{s}}^+)\Big] + \delta, \tag{3.39}$$

where $\hat{\mathbf{s}}^+ = \hat{\mathbf{s}}_Z - \boldsymbol{\mu}^+$, $\hat{\mathbf{s}}^- = \hat{\mathbf{s}}_Z - \boldsymbol{\mu}^-$, and $\delta = \log \frac{|\boldsymbol{\Sigma}^-|}{|\boldsymbol{\Sigma}^+|}$. During parameter optimization, we use $\boldsymbol{\Sigma}^{-1} = (\mathbf{L}^\top \mathbf{L})^{-1} = \mathbf{L}^{-1}(\mathbf{L}^{-1})^\top$ and directly infer $\mathbf{L}^{-1}$ to guarantee symmetric and positive semidefinit covariance matrices.

Beta Calibration

Recently, the authors in [43] introduced the Beta Calibration method which takes advantage of the fact that the confidence is defined in $[0, 1]$. Therefore, it is possible to directly rescale the confidence using beta distributions for $f_{\hat{P}}(\hat{p}|\bar{Y})$. The authors in [43] use the derivation for the calibrated confidence in (3.30) using the uncalibrated confidence $\hat{p} \in [0, 1]$ as function input and derive the log likelihood ratio

$$\ell r(\hat{p}) = \log\left[\frac{\mathrm{B}(\alpha^-, \beta^-)}{\mathrm{B}(\alpha^+, \beta^+)} \frac{\hat{p}^{\alpha^+ - 1}(1 - \hat{p})^{\beta^+ - 1}}{\hat{p}^{\alpha^- - 1}(1 - \hat{p})^{\beta^- - 1}}\right] \tag{3.40}$$

$$= a \cdot \log(\hat{p}) - b \cdot \log(1 - \hat{p}) + c, \tag{3.41}$$

between two beta distributions for $\bar{Y} = 1$ and $\bar{Y} = 0$, respectively, with the parameters $a = \alpha^+ - \alpha^-$, $b = \beta^- - \beta^+$, and $c = \log \frac{\mathrm{B}(\alpha^-, \beta^-)}{\mathrm{B}(\alpha^+, \beta^+)}$. Furthermore, $\mathrm{B}(\alpha, \beta)$ is the beta function. In practice, the parameters $a, b \in \mathbb{R}_{>0}$ and $c \in \mathbb{R}$ are estimated by MLE using the NLL loss [43].

Similar to the multivariate extension of the Logistic Calibration method for object detection calibration, we can derive a multivariate and conditional independent calibration mapping according to (3.32) for the Beta

Calibration method as well. Since Beta Calibration aims to directly rescale the confidences, we further use $\hat{\mathbf{S}} \in \mathcal{S}_{\hat{P}}$ as the surrogate for $\hat{\mathbf{S}}_{\hat{P}}$ for notational simplicity. Thus, the log likelihood ratio for the multivariate (independent) Beta Calibration is given by

$$\ell r(\hat{\mathbf{s}}) = \delta + \sum_{k=1}^{K} a_k \log(\hat{s}_k) - b_k \log(1 - \hat{s}_k), \tag{3.42}$$

where $a_k = \alpha_k^+ - \alpha_k^-$, $b_k = \beta_k^- - \beta_k^+$, and $\delta = \sum_{k=1}^{K} \log \frac{\mathrm{B}(\alpha_k^-, \beta_k^-)}{\mathrm{B}(\alpha_k^+, \beta_k^+)}$ with the multivariate beta function $\mathrm{B}(\boldsymbol{\alpha})$. Similar to the univariate case, we optimize $\mathbf{a}, \mathbf{b} \in \mathbb{R}_{>0}^K$ and $c \in \mathbb{R}$ in practice. The multivariate extension of the Beta Calibration method under the assumption of conditional dependence is not that straight forward since the natural multivariate extension of a beta distribution is a Dirichlet distribution $\mathrm{Dir}(\hat{\mathbf{s}}; \boldsymbol{\alpha})$ with shape parameters $\alpha \in \mathbb{R}_{>0}^K$. The Dirichlet distribution is defined for $\sum_{k=1}^{K} \hat{s}_k = 1$. However, this is not suitable in our case as we do not work with a multivariate probability vector as the input to the calibration function. The input $\hat{\mathbf{s}} \in \mathcal{S}$ is not restricted to a sum of 1, so that we propose to utilize a multivariate beta distribution that has been defined by the authors in [95] and is given by

$$f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}|\hat{M}) = \mathrm{B}(\boldsymbol{\alpha})^{-1} \cdot \frac{\prod_{k=1}^{K} \left[\lambda_k^{\alpha_k} (\hat{s}_k^*)^{\alpha_k + 1} \hat{s}_k^{-2} \right]}{\left[1 + \sum_{k=1}^{K} \lambda_k \hat{s}_k^* \right]^{\sum_{k=0}^{K} \alpha_k}}, \tag{3.43}$$

with shape parameters $\alpha_k, \beta_k \in \mathbb{R}_{>0}$ for all $k \in \{0, ..., K\}$ and the abbreviations $\lambda_k = \frac{\beta_k}{\beta_0}$ and $\hat{\mathbf{s}}^* = \frac{\hat{\mathbf{s}}}{1 - \hat{\mathbf{s}}}$. This allows for a derivation of a log likelihood ratio defined by

$$\begin{aligned} \ell r(\hat{\mathbf{s}}) = & \sum_{k=1}^{K} \left[\alpha_k^+ \log(\lambda_k^+) - \alpha_k^- \log(\lambda_k^-) + (\alpha_k^+ - \alpha_k^-) \log(\hat{s}_k^*) \right] + \\ & \sum_{k=0}^{K} \left[\alpha_k^- \log \left(\sum_{j=1}^{K} \lambda_j^- \hat{s}_j^* \right) - \alpha_k^+ \log \left(\sum_{j=1}^{K} \lambda_j^+ \hat{s}_j^* \right) \right] + \delta, \end{aligned} \tag{3.44}$$

where α^+, α^- and λ^+, λ^- denote the shape parameters for the positive and negative classes, respectively, and $c = \log \frac{\mathrm{B}(\mathbf{a}^-)}{\mathrm{B}(\mathbf{a}^+)}$.

3.4 Experiments for Semantic Confidence Calibration

In this section, we evaluate our calibration methods using different detection and segmentation architectures that are based on neural networks. We describe our experimental setup and show the calibration results for object detection, instance segmentation, and semantic segmentation. For each of these tasks, we use the MS COCO [44] and the Cityscapes [45] validation data sets. The respective data sets are divided into two equally sized parts for calibration training and evaluation, respectively. The respective training sets are used for the training of the forecaster itself, whereas no ground-truth label information are available for the respective test

sets, so that our experiments for calibration evaluation are limited to the validation sets. For Cityscapes, we use the Munster & Lindau images for calibration training and the Frankfurt images for evaluation, whereas the MS COCO validation set is split randomly. The experiments for multivariate confidence calibration evaluation have been subject of our publications in [1] and [4, p. 235 ff.].

Note that the MetaDetect framework by [38] (mentioned in Sec. 3.1) also applies an extended uncertainty evaluation of detection methods, which is related to our multivariate (conditional independent) calibration methods presented in Sec. 3.3.2. Similarly, the authors in [37] propose an equivalent approach for the uncertainty quantification within semantic segmentation. We leave a comparison of the works by [38] and [37] with our methods subject of future works.

3.4.1 Object Detection

We follow the experimental setup in [4, p. 235 f.] and evaluate the Histogram Binning [36], Logistic Calibration [35], and Beta Calibration [43] methods either using the confidence information only or by using all available information such as confidence, position, and shape of each detected object. We evaluate these methods on the MS COCO [44] and the Cityscapes [45] validation data sets that consist of 5.000 and 500 annotated images, respectively. For MS COCO, we use a pretrained `Faster R-CNN X101-FPN` [61] and a pretrained `RetinaNet R101-FPN` [9] model provided by [96] for inference. For the experiments on the Cityscapes data set, we use the bounding box information provided by a pretrained `Mask R-CNN R50-FPN` [31]. Samples with an uncalibrated confidence below 0.3 are neglected to keep the focus only on relevant detections. The experiments are performed for the classes *person*, *rider*, *car*, *truck*, *bus*, *train*, *motorcycle*, and *bicycle*, as these object categories are present in both data sets, which allows for a better comparison of the results. We further utilize the D-ECE, Brier score, and NLL as metrics for calibration evaluation as well as the Area under Precision-Recall Curve (AUPRC) to measure the model's detection performance. We divide our examinations into standard calibration evaluation where only the confidence information is used and into the multivariate calibration where the confidence in conjunction with all available bounding box information is used for calibration. For D-ECE calculation in the confidence-only case, a binning scheme with $I = 20$ equally sized bins is used over the confidence space. In contrast, the D-ECE for the multivariate case uses $I_k = 5$ bins for each dimension $k \in \{1, \dots, K\}$ which yields a total amount of $3,125$ bins. Uninformative bins with less than 8 samples are neglected during D-ECE computation. For the computation of the D-ECE, it is necessary to define the features that are used for the underlying binning scheme. Thus, comparing D-ECE scores obtained by different subsets of features is not applicable since the conditional probability distributions of $\hat{P}$ differ from each other. For example, a D-ECE score obtained by using the confidence, c_x, and c_y position should not be compared to a D-ECE score that is based on the confidence, width, and height, as the conditional distributions for the confidence, and thus the approximate binning schemes, differ from each other.

According to our definition for object detection calibration in (3.11), the calibration target is the precision, which depends on the IoU score used to distinguish between correctly predicted objects and false nega-

Table 3.1: Calibration results for object detection where only the predicted confidence $\hat{P}$ is used for calibration and D-ECE evaluation. We observe that the scaling methods Logistic Calibration and Beta Calibration consistently achieve the best calibration results for D-ECE, Brier score, and NLL. In contrast to Histogram Binning, the scaling methods apply a monotonically increasing mapping of uncalibrated confidence estimates to calibrated ones which does not affect the AUPRC [4, p. 238, Tab. 1].

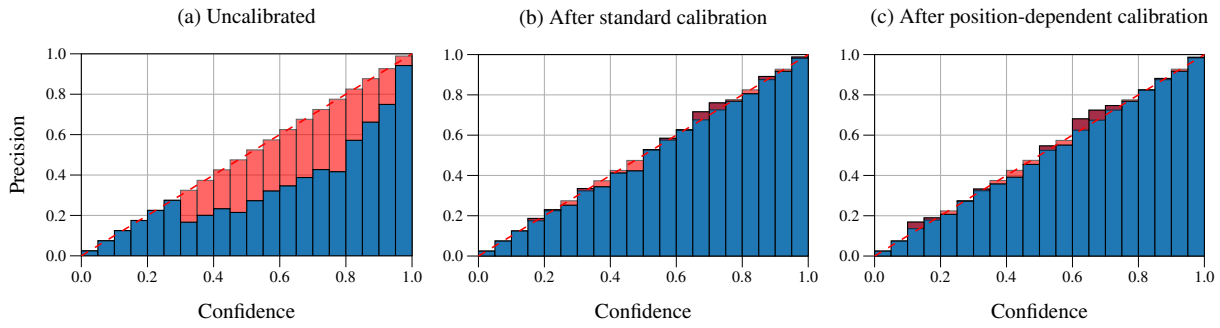
Network	IoU	Calibration method	D-ECE	Brier	NLL	AUPRC
`Faster R-CNN` (trained on MS COCO)	0.50	Uncalibrated	0.153	0.176	0.536	0.920
		Histogram Binning	0.026	0.146	0.470	0.878
		Logistic Calibration	0.021	**0.142**	**0.433**	0.920
		Beta Calibration	**0.020**	**0.142**	**0.433**	0.920
	0.75	Uncalibrated	0.294	0.257	0.829	0.866
		Histogram Binning	**0.026**	0.155	0.510	0.770
		Logistic Calibration	0.027	**0.144**	**0.448**	0.866
		Beta Calibration	0.030	**0.144**	0.449	0.866
`RetinaNet` (trained on MS COCO)	0.50	Uncalibrated	0.083	0.157	0.478	0.907
		Histogram Binning	0.025	0.152	0.482	0.889
		Logistic Calibration	0.024	**0.150**	**0.451**	0.907
		Beta Calibration	**0.022**	**0.150**	**0.451**	0.907
	0.75	Uncalibrated	0.151	0.172	0.518	0.855
		Histogram Binning	0.030	0.142	0.457	0.832
		Logistic Calibration	0.032	**0.140**	0.439	0.855
		Beta Calibration	**0.026**	**0.140**	**0.437**	0.855
`Mask R-CNN` (trained on Cityscapes)	0.50	Uncalibrated	0.108	0.145	0.496	0.952
		Histogram Binning	0.033	0.133	0.493	0.902
		Logistic Calibration	**0.029**	**0.124**	**0.378**	0.952
		Beta Calibration	**0.029**	0.125	0.379	0.952
	0.75	Uncalibrated	0.296	0.269	1.055	0.896
		Histogram Binning	**0.036**	0.160	0.547	0.757
		Logistic Calibration	0.042	**0.135**	**0.421**	0.896
		Beta Calibration	0.044	**0.135**	0.422	0.896

tives. Thus, we run our evaluations using an IoU threshold of 0.50 and 0.75. The calibration results for the confidence-only case as well as for the multivariate calibration case are given in Tab. 3.1 and Tab. 3.2, respectively. For further insights, we show the reliability diagrams for all calibration cases in Fig. 3.4.

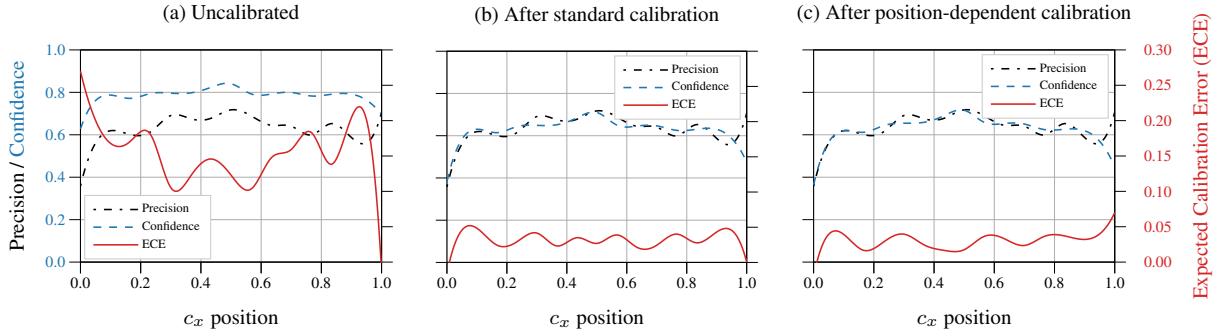
For each inspected object detection model, we observe a high deviation between predicted confidence and observed precision which is indicated by high scores, especially for D-ECE but also for Brier score and NLL. This holds for the confidence-only case as well as for the position-dependent case. The reliability diagrams in Fig. 3.4i reveal an overconfidence of the `Faster R-CNN` on the MS COCO data set. This is in agreement with the current state of research which evaluates common neural network architectures as overconfident [13, 78]. In contrast, the `RetinaNet` architecture is trained using the focal loss [9] which is known of resulting in low confidence estimates and thus in underconfidence [77, 2]. In both cases, the object detection models are miscalibrated which is alleviated using confidence calibration. In our experiments, we

Table 3.2: Calibration results for object detection where all information $\hat{\mathbf{S}} = (\hat{P}, \hat{Y}, \hat{\mathbf{R}})^\top$ are used for confidence calibration and D-ECE evaluation. In this case, $\hat{\mathbf{R}}$ denotes the bounding box encoding that consists of the center x and y positions c_x, c_y as well as of the width w and height h. Similar to the results for the confidence-only case in Tab. 3.1, the scaling methods consistently achieve the best results. However, the multivariate confidence calibration is not a monotonically increasing function any more and thus has an influence on the AUPRC, which is marginal in this case [4, p. 239, Tab. 2].

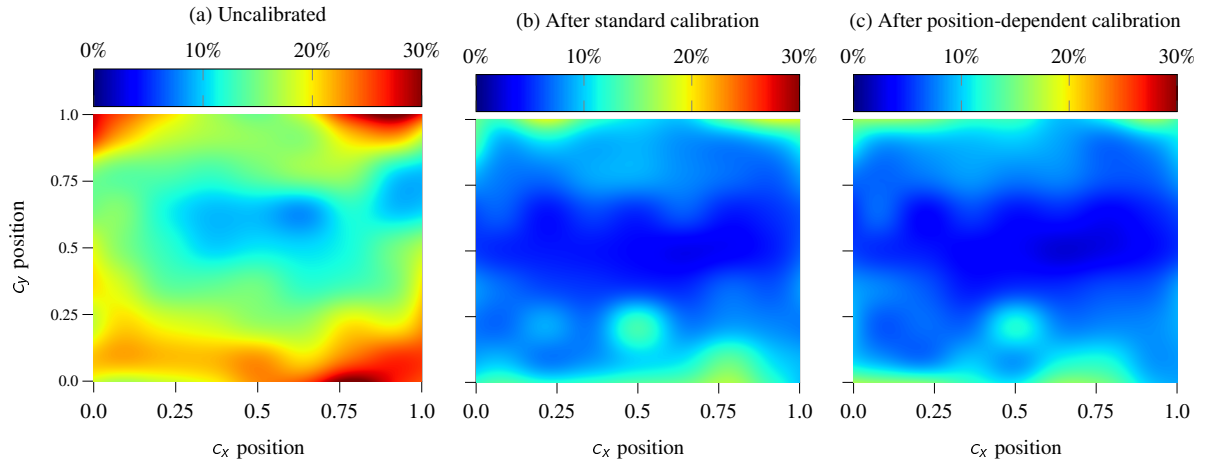
Network	IoU	Calibration method	D-ECE	Brier	NLL	AUPRC
`Faster R-CNN` (trained on MS COCO)	0.50	Uncalibrated	0.119	0.176	0.536	**0.920**
		Histogram Binning	0.052	0.174	0.712	0.829
		Logistic Calibration (independent)	**0.041**	**0.143**	**0.436**	0.919
		Logistic Calibration (dependent)	0.043	0.146	0.456	0.915
		Beta Calibration (independent)	0.042	0.145	0.442	0.916
		Beta Calibration (dependent)	0.046	0.146	0.444	0.914
	0.75	Uncalibrated	0.227	0.257	0.829	**0.866**
		Histogram Binning	0.059	0.186	0.689	0.723
		Logistic Calibration (independent)	**0.044**	**0.145**	**0.452**	0.864
		Logistic Calibration (dependent)	0.047	0.149	0.469	0.856
		Beta Calibration (independent)	0.047	0.146	0.454	0.862
		Beta Calibration (dependent)	0.047	0.147	0.456	0.861
`RetinaNet` (trained on MS COCO)	0.50	Uncalibrated	0.072	0.157	0.478	0.907
		Histogram Binning	0.046	0.175	0.739	0.842
		Logistic Calibration (independent)	**0.045**	**0.149**	**0.450**	**0.908**
		Logistic Calibration (dependent)	0.049	0.153	0.474	0.903
		Beta Calibration (independent)	0.046	0.150	0.458	0.906
		Beta Calibration (dependent)	0.053	0.155	0.467	0.901
	0.75	Uncalibrated	0.110	0.172	0.518	**0.855**
		Histogram Binning	0.049	0.162	0.668	0.756
		Logistic Calibration (independent)	0.048	0.140	**0.439**	**0.855**
		Logistic Calibration (dependent)	0.048	0.142	0.463	0.848
		Beta Calibration (independent)	**0.047**	**0.139**	**0.439**	0.853
		Beta Calibration (dependent)	0.053	0.144	0.450	0.844
`Mask R-CNN` (trained on Cityscapes)	0.50	Uncalibrated	0.102	0.145	0.496	**0.952**
		Histogram Binning	0.053	0.150	0.536	0.857
		Logistic Calibration (independent)	0.038	**0.125**	**0.381**	0.950
		Logistic Calibration (dependent)	0.045	0.133	0.437	0.948
		Beta Calibration (independent)	**0.036**	0.127	0.404	0.950
		Beta Calibration (dependent)	0.063	0.134	0.413	0.925
	0.75	Uncalibrated	0.281	0.269	1.055	0.896
		Histogram Binning	0.080	0.194	0.606	0.685
		Logistic Calibration (independent)	0.056	**0.135**	**0.424**	**0.901**
		Logistic Calibration (dependent)	0.064	0.139	0.462	0.896
		Beta Calibration (independent)	**0.052**	0.137	0.503	0.895
		Beta Calibration (dependent)	0.096	0.160	0.491	0.832



(i) Reliability diagrams w.r.t. the confidence only [4, p. 237, Fig. 3].



(ii) Reliability diagrams w.r.t. the c_x position of the predicted objects (1d) [4, p. 240, Fig. 4].



(iii) Reliability diagrams w.r.t. the c_x and c_x position of the predicted objects (2d) [4, p. 240, Fig. 5].

Figure 3.4: Reliability diagrams (object detection) for a `Faster R-CNN` on the MS COCO calibration validation set for class *pedestrian* [4, p. 237 ff., Fig. 3-5] with uncalibrated confidences $\hat{P} \geq 0.3$. The uncalibrated baseline model is consistently overconfident for all confidence levels with increasing miscalibration towards the image boundaries. In this example, standard calibration shows good performance which is slightly improved using position-dependent calibration.

found that especially the scaling methods Logistic Calibration and Beta Calibration are able to successfully recalibrate the detection models. Furthermore, these methods provide a monotonically increasing mapping of uncalibrated confidences to calibrated one in the standard (confidence only) case and therefore do not affect the AUPRC scores. In contrast, Histogram Binning has no restrictions and thus leads to a degradation of the prediction performance.

Similar to the standard (confidence only) calibration, our position-dependent calibration and especially the scaling methods are also able to provide a meaningful recalibration mapping. The reliability diagrams show minor improvements in the position-dependent calibration of the models especially in the 2d case. In contrast to confidence-only calibration, the position-dependent calibration is not a monotonically increasing mapping and thus does affect the AUPRC scores. However, the effect on the AUPRC scores is marginal for the scaling methods. By comparing the conditional independent calibration methods with their conditional dependent counterparts, we observe no improvements in calibration and only minor differences in the results. We could find a low connection between position and miscalibration which, however, is not as strong as initially suggested. This might be the reason why the conditional dependent methods do not yield further improvements in calibration in our experiments. Therefore, we conclude that the standard scaling methods already provide sufficient confidence calibration, which is further improved by our conditionally independent position-dependent calibration methods. This is a valuable extension especially for safety-critical applications and subsequent processes which will be further investigated in Chap. 6.

3.4.2 Instance Segmentation

The experiments shown here are part of our previous work in [4, p. 239 ff.]. For the evaluation of calibration within the task of instance segmentation, we apply a pretrained `Mask R-CNN` [31] as well as a pretrained `PointRend` [97] on the MS COCO and Cityscapes validation data sets. Since the target is to perform recalibration for the instance segmentation masks, we can use each pixel as an own input to the calibration functions. This leads to a large training data set (e.g., approx. 45 million samples for the class *pedestrian* within the Cityscapes data set), resulting in a data set that is too large to train the scaling methods on common hardware in a reasonable time. Furthermore, it has recently been shown that binning methods such as Histogram Binning yield a more robust calibration mapping compared to scaling methods given a large amount of data [88]. In contrast, a large data set is available for the task of instance segmentation calibration. Therefore, we only use the Histogram Binning as calibration method for instance segmentation calibration.

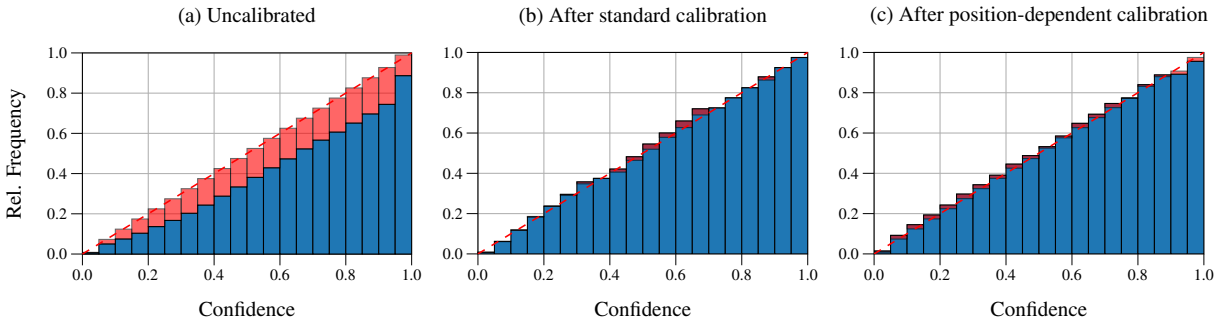
Instance segmentation is a joint task of object detection and semantic segmentation within a predicted bounding box. Similar to object detection, it is necessary to distinguish between true and false positives which requires a certain IoU threshold. Thus, we run our evaluations using an IoU threshold of 0.50 and 0.75, respectively. Similar to our experiments for object detection, we further investigate the effect of standard (confidence-only) calibration as well as of position-dependent calibration. We use the pixel x and y position as position information which are normalized to the bounding box size to get $x, y \in [0, 1]$ for calibration. Furthermore, we suspect a correlation between miscalibration and a pixel's distance to the next segment

Table 3.3: Calibration results for instance segmentation where only the predicted pixel confidence $\hat{P}_j$ is used for calibration and D-ECE evaluation [4, p. 242, Tab. 3-4]. The Histogram Binning is able to reduce miscalibration of the mask scores while preserving the mask quality in all cases.

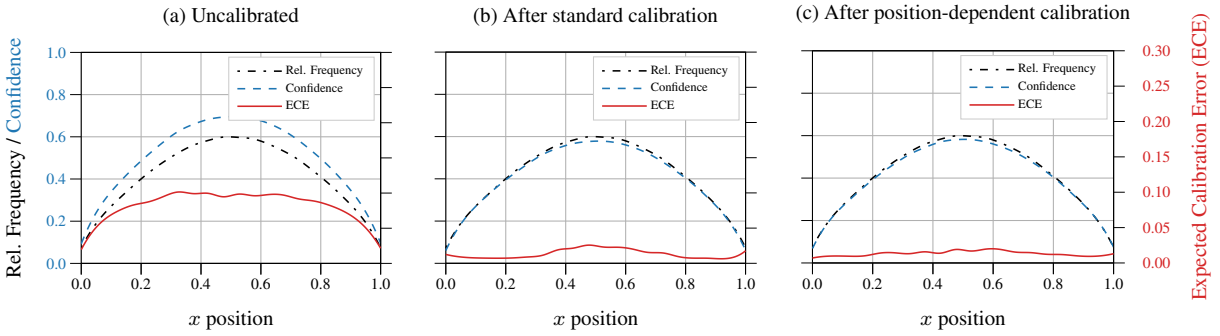
Network	Data set	IoU	Calibration method	D-ECE	Brier	NLL	AUPRC
Mask R-CNN	Cityscapes	0.50	Uncalibrated	0.071	0.110	0.432	**0.724**
			Histogram Binning	**0.057**	**0.099**	**0.320**	0.723
		0.75	Uncalibrated	0.129	0.147	0.622	**0.375**
			Histogram Binning	**0.059**	**0.108**	**0.340**	**0.375**
	MS COCO	0.50	Uncalibrated	0.220	0.222	0.940	**0.663**
			Histogram Binning	**0.064**	**0.150**	**0.442**	0.662
		0.75	Uncalibrated	0.266	0.250	1.070	**0.237**
			Histogram Binning	**0.060**	**0.144**	**0.423**	0.235
PointRend	Cityscapes	0.50	Uncalibrated	0.129	0.160	0.785	**0.709**
			Histogram Binning	**0.027**	**0.105**	**0.326**	0.698
		0.75	Uncalibrated	0.187	0.192	0.929	**0.347**
			Histogram Binning	**0.039**	**0.115**	**0.349**	0.344
	MS COCO	0.50	Uncalibrated	0.223	0.222	0.946	**0.672**
			Histogram Binning	**0.063**	**0.144**	**0.428**	0.664
		0.75	Uncalibrated	0.266	0.248	1.060	**0.258**
			Histogram Binning	**0.067**	**0.138**	**0.411**	0.238

Table 3.4: Calibration results for instance segmentation where all information $\hat{\mathbf{S}} = (\hat{P}_j, \hat{Y}, \hat{\mathbf{R}}_j)^\top$ are used for confidence calibration and D-ECE evaluation [4, p. 242, Tab. 3-4]. The multivariate Histogram Binning also reduces miscalibration. Furthermore, it significantly improves the mask quality.

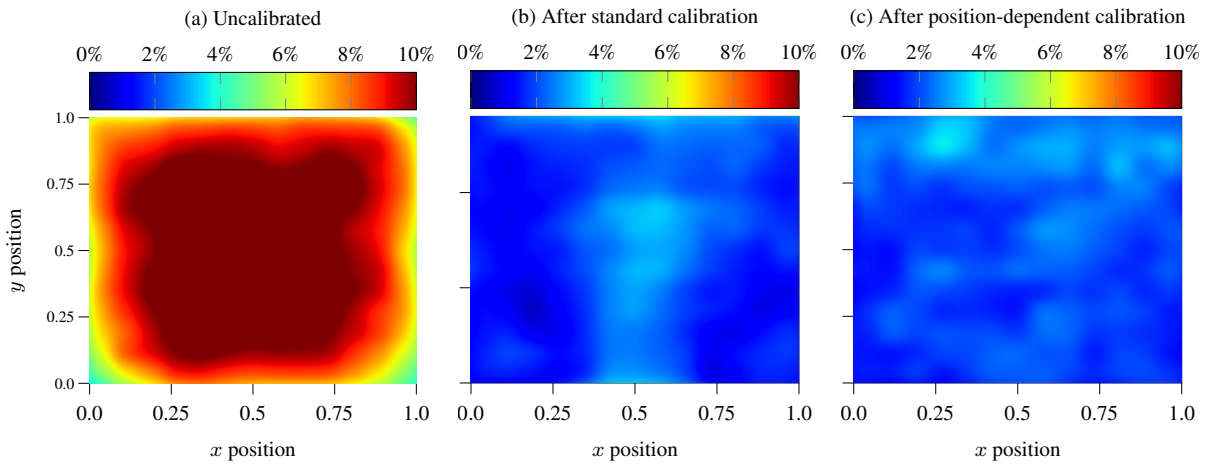
Network	Data set	IoU	Calibration method	D-ECE	Brier	NLL	AUPRC
Mask R-CNN	Cityscapes	0.50	Uncalibrated	0.112	**0.110**	**0.432**	0.724
			Histogram Binning	**0.101**	0.117	0.530	**0.787**
		0.75	Uncalibrated	0.145	0.147	0.622	0.375
			Histogram Binning	**0.103**	**0.120**	**0.523**	**0.479**
	MS COCO	0.50	Uncalibrated	0.234	0.222	0.940	0.663
			Histogram Binning	**0.136**	**0.171**	**0.776**	**0.760**
		0.75	Uncalibrated	0.272	0.250	1.070	0.237
			Histogram Binning	**0.129**	**0.165**	**0.720**	**0.425**
PointRend	Cityscapes	0.50	Uncalibrated	0.209	**0.160**	**0.785**	0.709
			Histogram Binning	**0.190**	0.190	1.299	**0.758**
		0.75	Uncalibrated	0.254	0.192	**0.929**	0.347
			Histogram Binning	**0.184**	**0.191**	1.247	**0.486**
	MS COCO	0.50	Uncalibrated	0.240	0.222	**0.946**	0.672
			Histogram Binning	**0.161**	**0.180**	1.005	**0.751**
		0.75	Uncalibrated	0.274	0.248	1.060	0.258
			Histogram Binning	**0.153**	**0.173**	**0.936**	**0.388**



(i) Reliability diagrams w.r.t. the confidence only [4, p. 243, Fig. 6].



(ii) Reliability diagrams w.r.t. the relative x position of each mask pixel (1d) [4, p. 243, Fig. 7].



(iii) Reliability diagrams w.r.t. the relative x and y position of each mask pixel (2d) [4, p. 243, Fig. 8].

Figure 3.5: Reliability diagrams (instance segmentation) for a `Mask R-CNN` on the MS COCO calibration validation set for class *pedestrian* [4, p. 243, Fig. 6-8]. The uncalibrated pixel confidences are consistently too overconfident for all confidence levels. Furthermore, the gap between predicted pixel confidence and observed frequency increases towards the mask's center. This is mitigated by standard calibration as well as by our extended methods.

boundary. Thus, we also use the (normalized) distance as an additional feature for calibration. For calibration evaluation, we use the D-ECE with $I_k = 15$ bins for each dimension $k \in \{1, \dots, K\}$. Furthermore, the Brier and NLL scores are used as complementary evaluation metrics. Finally, we report the Mean Intersection over Union (mIoU) which is the mean IoU score over all classes and which is an indicator for the quality of the predicted instance segmentation masks. The evaluation results for the confidence-only case as well as for the multivariate calibration case are given in Tab. 3.3 and Tab. 3.4, respectively. For further insights, we show the reliability diagrams for the confidence only as well as for including position information in Fig. 3.5.

As visualized in Fig. 3.5, the instance segmentation models are consistently overconfident in their predictions for the pixel confidence. This miscalibration is reduced by the standard Histogram Binning as well as by the position-dependent Histogram Binning. In some cases, the position-dependent Histogram Binning does not lead to an improvement in the complementary Brier and NLL scores. On the one hand, we can observe a strong connection between position information and miscalibration. In this case, both calibration schemes lead to an improvement in calibration, whereas the position-dependent variant results in a more uniform calibration over the x and y space compared to its confidence-only counterpart. On the other hand, our experiments show that position-dependent calibration is able to significantly improve the quality of the segmentation masks which is indicated by the gain in the mIoU scores for all evaluated models. For standard calibration, the mask scores are only rescaled by their confidence which might lead to a better calibration but sometimes also to unwanted losses of mask segments (especially small objects in the background). In contrast, position-dependent calibration is able to apply a recalibration that is also aware of possible correlations between pixel confidence and object size. We assume that this leads to improved estimates of the mask confidences even for smaller objects. Therefore, we conclude that especially the position-dependent calibration is a valuable contribution towards reliable confidence information and improved segmentation masks for the task of instance segmentation.

3.4.3 Semantic Segmentation

As opposed to instance segmentation, it is not necessary to identify single objects within semantic segmentation but to determine the membership of each image pixel to a general class. Therefore, the experiments for semantic segmentation calibration do not rely on a preceding detection stage and we can use each image pixel as an input for calibration training and evaluation using the Histogram Binning. We use a pretrained `DeepLabv3+` [33] and a pretrained `DeepLabv2` [32] on the Cityscapes and MS COCO validation data sets, respectively, as well as a pretrained `HRNet` [98, 99, 100]. We further use the same additional features for confidence calibration such as relative x, y position and the pixel's distance to the next segment boundary. The calibration results are shown in Tab. 3.5 for the confidence-only calibration case as well as in Tab. 3.6 for the position-dependent case. Furthermore, we show the respective reliability diagrams in Fig.3.6 to gain further insights in the calibration properties of the examined models. In contrast to instance segmentation, the examined semantic segmentation models provide already well-calibrated confidence estimates on pixel-level. In this case, neither standard Histogram Binning nor our position-dependent calibration are able to further improve the calibration. However, position-dependent calibration leads to a slight degradation of the mask

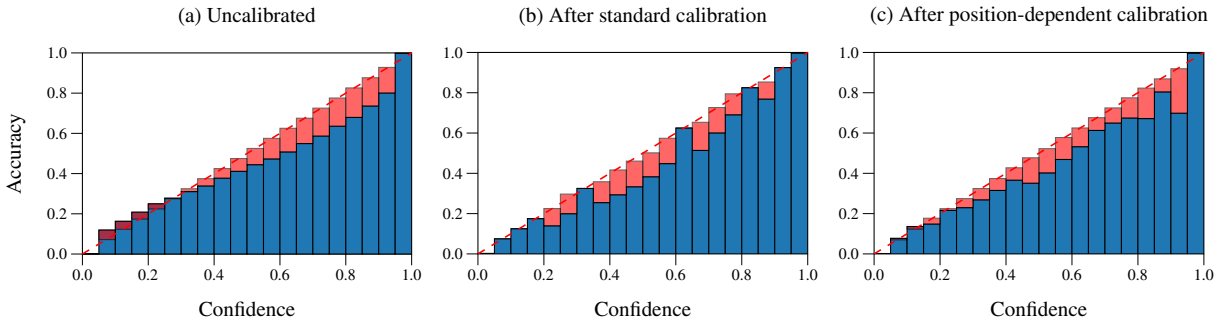
Table 3.5: Calibration results for semantic segmentation where only the predicted pixel confidence $\hat{P}_j$ is used for calibration and D-ECE evaluation [4, p. 245, Tab. 5]. The uncalibrated segmentation models are already well-calibrated. The Histogram Binning does not affect the mask quality and only leads to minor improvements in confidence calibration.

Network	Data set	Calibration Method	D-ECE	Brier	NLL	mIoU
DeepLabv3+	Cityscapes	Uncalibrated	0.0016	0.060	**0.139**	**0.623**
		Histogram Binning	**0.0008**	0.060	0.170	0.619
DeepLabv2	MS COCO	Uncalibrated	0.0009	0.458	**1.173**	**0.933**
		Histogram Binning	**0.0006**	**0.456**	1.515	**0.933**
HRNet	Cityscapes	Uncalibrated	**0.0007**	0.057	**0.115**	**0.629**
		Histogram Binning	0.0008	0.057	0.148	0.628
	MS COCO	Uncalibrated	0.0046	0.779	5.812	**0.939**
		Histogram Binning	**0.0006**	**0.563**	**2.261**	**0.939**

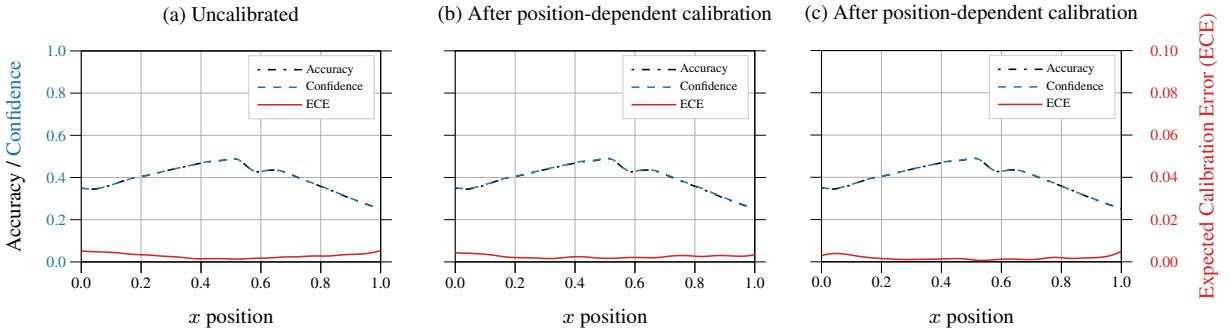
Table 3.6: Calibration results for semantic segmentation where all information $\hat{\mathbf{S}} = (\hat{P}_j, \hat{\mathbf{R}}_j)^\top$ are used for confidence calibration and D-ECE evaluation [4, p. 245, Tab. 5]. In contrast to the case in Tab. 3.5, the position-dependent calibration is not able to improve calibration. Furthermore, it leads to a minor degradation of the mask quality as indicated by the mIoU.

Network	Data set	Calibration Method	D-ECE	Brier	NLL	mIoU
DeepLabv3+	Cityscapes	Uncalibrated	0.0019	**0.060**	**0.139**	**0.623**
		Histogram Binning	0.0019	0.062	0.189	0.589
DeepLabv2	MS COCO	Uncalibrated	0.0015	**0.458**	**1.173**	**0.933**
		Histogram Binning	0.0015	0.485	1.790	0.913
HRNet	Cityscapes	Uncalibrated	**0.0015**	**0.057**	**0.115**	**0.629**
		Histogram Binning	0.0019	0.060	0.171	0.582
	MS COCO	Uncalibrated	0.0046	0.779	0.812	**0.939**
		Histogram Binning	**0.0014**	**0.571**	**0.372**	0.931

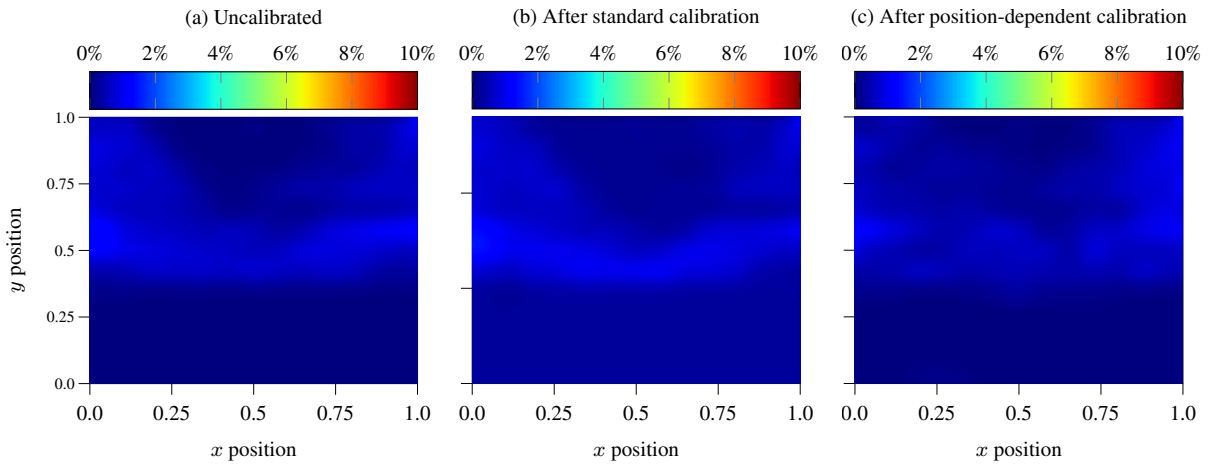
quality as stated by the mIoU scores. Although the reliability diagrams in Fig. 3.6i show an overconfidence for the confidence range $P \in [0.35, 0.95]$, most pixels have a low confidence of $P < 0.1$, which results in an overall low miscalibration score (D-ECE). We suspect that the major difference between instance and semantic segmentation in calibration in our experiments arises from the different approaches for model training. As already pointed out, instance segmentation is a joint task of object detection and segmentation. Thus, the quality of the predicted bounding boxes directly affects the performance of the segmentation head. It is a more challenging task to identify single objects in an image and to jointly optimize the network for object detection and segmentation. Furthermore, a semantic segmentation model uses the whole image for model training and thus has more pixels available, whereas an instance segmentation model is restricted to the pixels within a detected bounding box. Therefore, we conclude that the semantic segmentation models, that have been investigated in our experiments, already provide well-calibrated confidence estimates and thus do not require an additional post-hoc calibration step.



(i) Reliability diagrams w.r.t. the confidence only [4, p. 246, Fig. 11].



(ii) Reliability diagrams w.r.t. the relative x position of each mask pixel (1d) [4, p. 247, Fig. 12].



(iii) Reliability diagrams w.r.t. the relative x and y position of each mask pixel (2d) [4, p. 247, Fig. 13].

Figure 3.6: Reliability diagrams (semantic segmentation) for a `DeepLabv3+` on the Cityscapes calibration validation set for class *pedestrian* [4, p. 246 f., Fig. 11-13]. The uncalibrated segmentation model already provides well-calibrated confidences (most samples are located at confidence levels of 0 or 1). This can be also seen in the 1d and 2d reliability diagrams. Thus, neither standard nor position-dependent calibration lead to significant changes in calibration.

3.5 Conclusion for Semantic Confidence Calibration

Object detection is the joint task of predicting the position, shape, and class of individual objects. For each predicted object, a confidence score is estimated for the predicted class that can be interpreted as a probability of the correctness of the predicted class label. In this chapter, we examined this semantic confidence score for consistency, i.e., if the estimated uncertainty corresponds to the observed error. We started by deriving the definitions of semantic confidence calibration for the tasks of object detection, instance segmentation, and semantic segmentation. These definitions allow to investigate if the position information has an influence on the semantic confidence as well. This also holds for instance and semantic segmentation where the position of each pixel might have an influence to the confidence. Thus, these definitions allow for an extension of the common Expected Calibration Error (ECE) metric to the task of object detection and segmentation. We used the new Detection ECE (D-ECE) to measure the semantic miscalibration of detection and segmentation models.

We extended the common calibration methods Histogram Binning [36], Logistic Calibration [35], and Beta Calibration [43] to include additional information such as position and shape into a calibration and to capture possible correlations between the given information. These multivariate calibration methods can be used to apply a post-hoc calibration of the confidence scores provided by a detection or segmentation model. The multivariate scaling methods Logistic Calibration and Beta Calibration calibration are further divided into conditionally independent and dependent variants. Both variants use the confidence and the position of the detected objects for confidence recalibration. Furthermore, both variants are able to model an influence of the additional position information to the calibration result. However, the conditionally independent scaling methods assume that the input random variables follow independent probability distributions. In contrast, the conditionally dependent methods are able to capture possible dependencies in the input.

The proposed multivariate calibration methods have been evaluated using different neural network architectures and different data sets. Our investigations show that in our experiments, the semantic segmentation models are already well calibrated so that our calibration methods have not been able to gain further improvements. In contrast, the multivariate confidence calibration has a positive effect on the calibration properties of object detection and especially on instance segmentation models. Although we could only find a minor connection between position and miscalibration, our extended methods show a qualitatively good calibration performance. In the case of instance segmentation, the extended multivariate calibration is able to not only improve the calibration properties of the model but also to enhance the model's prediction performance. We suspect that the major difference between instance and semantic segmentation is a result of the different training procedures that are used for model training.

Therefore, we conclude that our multivariate extension of the calibration methods provide a powerful framework for the recalibration of semantic uncertainty. Since object detection is a part of the image-based environment perception process (cf. Chap. 1), we further investigate the effect of semantic confidence calibration on object detection within a subsequent object tracking in Chap. 6.

4 Bayesian Confidence Calibration

In the previous chapter 3, we derived the definitions for semantic confidence calibration from simple classification to the more complex detection, instance, and semantic segmentation tasks. Furthermore, we proposed methods for a position-dependent confidence calibration. But what if the calibration method itself is uncertain in some cases, e.g., for an input sample which falls into a region with a sparsely populated training set? For example, if a calibration method is embedded into a safety-relevant context, it might still lead to a false sense of safety if applied to situations which are unknown either for the baseline model or for the calibration mapping. In this case, it is advantageous to get a self-assessment about the epistemic calibration model uncertainty. In contrast to the examinations for the aleatoric uncertainty given by the detector confidence, we seek for the epistemic uncertainty which is inherent in a calibration function itself.

Therefore, we introduce the term of Bayesian confidence calibration which adapts the idea of probabilistic modeling similar to Bayesian neural networks. Within a Bayesian neural network, the weights in the network layers are treated as probability distributions that express the uncertainty in the weights [58, 101]. In this way, it is possible to sample multiple realizations of the network weights from these distributions. Each of these weight combinations is then used to generate a sample distribution for each output. In practice, it is possible to approximate such a Bayesian neural network using dropout during the inference which is also known as Monte-Carlo dropout [58].

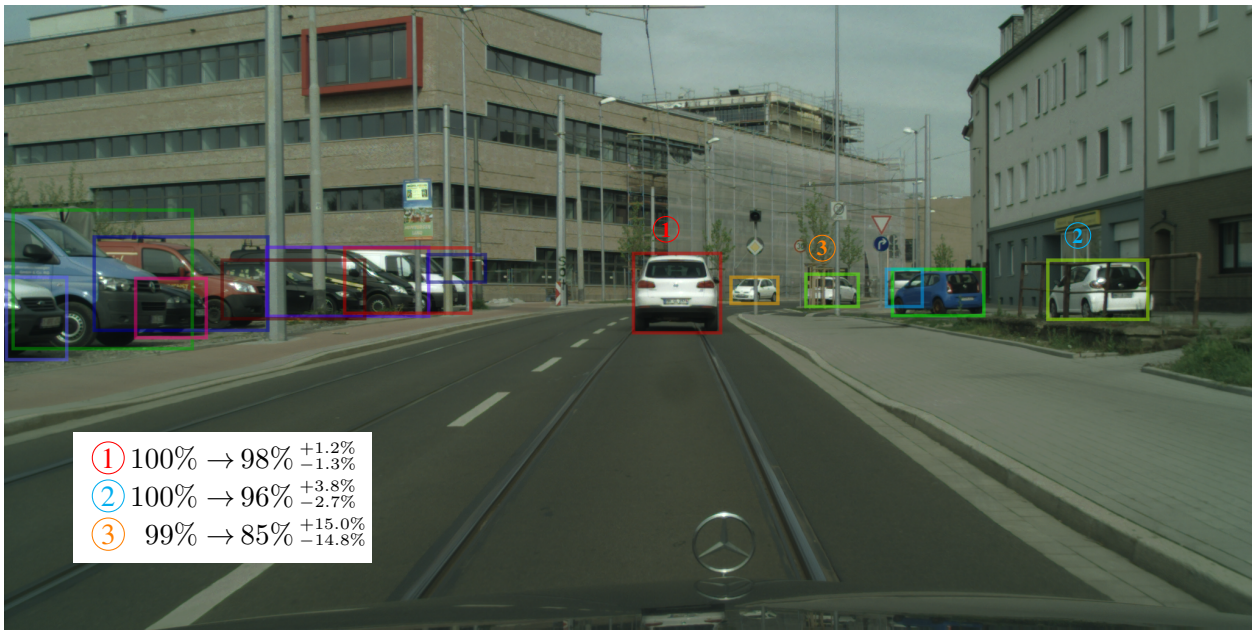


Figure 4.1: Qualitative example of Bayesian confidence calibration for object detection [3, p. 1, Fig. 1]. For each detection, a calibration method reassigns a new confidence score. If we apply Bayesian confidence calibration, it is possible to yield a prediction interval for the calibrated confidence to express the epistemic uncertainty in calibration.

We adapt this idea of modeling the epistemic uncertainty and seek to transfer this to our calibration functions. Similarly, a probability distribution is placed over each calibration weight. However, in contrast to Bayesian neural networks, we can not use Monte-Carlo dropout for calibration methods as these functions do not use any dropout or only use a few weights for recalibration, e.g., Temperature Scaling [13]. Therefore, we approximate the weight distributions using Stochastic Variational Inference (SVI) which allows for training variational distributions for each calibration weight. In this way, it is possible to obtain multiple calibrated confidence estimates for a single input sample which, in turn, describe a sample distribution and thus reflect the epistemic uncertainty of the calibration mapping itself. The concept of Bayesian confidence calibration is qualitatively shown in Fig. 4.1. The work presented in this chapter was subject of the publication in [3].

In Sec. 4.1, we derive the Bayesian confidence calibration and describe how to obtain the calibrated confidence estimates in conjunction with its epistemic uncertainty. The Bayesian calibration framework is afterwards used for the evaluation of the calibration performance and of the quality of the provided epistemic uncertainty in Sec. 4.2. In our studies, we focus on the task of object detection since the Bayesian confidence calibration framework requires a sampling during inference which has been computational too expensive for the application to instance or semantic segmentation. We give a conclusion about the methods and our findings in Sec. 4.3.

As opposed to Chap. 3 and Chap. 5, we do not aim to integrate this Bayesian framework into a subsequent process such as object tracking in this work. Our target is to show a possibility to capture additional model uncertainties to increase the awareness of possible failure modes which is important for safety-critical applications. Nevertheless, the integration of Bayesian confidence calibration, e.g., into an object tracking process, is an interesting use case which we let open for future work.

Contributions: Our contributions are summarized by:

- Definition of Bayesian confidence calibration.
- Methods for Bayesian confidence calibration.

4.1 Epistemic Uncertainty Modeling of Confidence Calibration

We use the preceding notation and follow the definition of confidence calibration for classification in (3.1) as well as for object detection in (3.11). In this scope, we work with an arbitrary classification or detection model that estimates a label $\hat{Y} \in \mathcal{Y}$ as well as a confidence $\hat{P} \in [0, 1]$ for each prediction where the confidence expresses the model's belief about the correctness of the actual prediction. We denote this belief as $\hat{M} \sim \text{Bern}(\hat{P})$. An object detection model also outputs a position and shape estimate $\hat{\mathbf{R}} \in \mathcal{R}$ for each predicted objects, so that the joint conditioned model distribution is denoted by $f_{\hat{P},\hat{Y},\hat{\mathbf{R}}}(\hat{p}, \hat{y}, \hat{\mathbf{r}}|\mathbf{x})$. We further denote $\hat{\mathbf{S}} = (\hat{P}, \hat{Y}, \hat{\mathbf{R}})^\top \in \mathcal{S}$[1] as the aggregated variable for the model output. These predictions aim

[1] In the preceding chapter, we distinguished between $\mathcal{S}_P$ as the aggregated output space using the confidence $\hat{P}$, and $\mathcal{S}_Z$ as the output space using the raw network logits Z. Since the derivation for Bayesian confidence calibration holds for both cases, we use the shorthand notation of $\mathcal{S}$ to represent either $\mathcal{S}_P$ or $\mathcal{S}_Z$.

to target the true label $\bar{Y} \in \mathcal{Y}$ and the true position $\bar{\mathbf{R}} \in \mathcal{R}$ which follow the joint ground-truth data distribution $f_{\mathbf{X},\bar{Y},\bar{\mathbf{R}}}(\mathbf{x}, \bar{y}, \bar{\mathbf{r}}) = f_{\bar{Y},\bar{\mathbf{R}}}(\bar{y}, \bar{\mathbf{r}}|\mathbf{x}) f_{\mathbf{X}}(\mathbf{x})$. For confidence calibration, we adapt the scaling methods presented in Sec. 3.3.2 whose calibration parameters $\boldsymbol{\theta} \in \Theta$ are commonly obtained by Maximum Likelihood Estimation (MLE) using a data set $\mathcal{D} = \{(\hat{\mathbf{s}}_n, \hat{m}_n)\}_{n=1}^{N}$ with N samples, where Θ denotes the set of all possible parameters. A position-dependent calibration function $h_{\boldsymbol{\theta}} : \mathcal{S} \to [0, 1]$ serves as a mapping from the uncalibrated confidence, the estimated label and the predicted position to a calibrated confidence estimate $\hat{Q} \in [0, 1]$, so that $\hat{Q} = h_{\boldsymbol{\theta}}(\hat{\mathbf{S}})$. The calibrated confidence is then used as the Bernoulli parameter for $\hat{M}$, so that $\hat{M} \sim \text{Bern}(\hat{Q})$.

Similar to neural networks, a calibration mapping may also exhibit epistemic uncertainty, e.g., due to an insufficient amount of training data (cf. Sec. 2.3). Thus, we are interested in modeling the epistemic uncertainty of the calibrated confidence $\hat{Q}$ for a new sample $\hat{\mathbf{s}}^* \in \mathcal{S}$ during inference given the calibration parameters $\boldsymbol{\theta}$ and the calibration training data set $\mathcal{D}$. Therefore, we do not interpret the calibrated confidence as a deterministic distribution parameter but rather as a random variable that follows a certain probability distribution $\hat{Q}|\hat{\mathbf{s}}^*, \mathcal{D} \sim f_{\hat{Q}}$. In Bayesian statistics, we can infer the conditional distribution of $\hat{Q}$ using the posterior predictive distribution that denotes the probability of $\hat{Q}$ weighted by the posterior for the calibration parameters $f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D})$. The posterior for the calibration parameters is marginalized out so that the posterior predictive distribution is given by

$$f_{\hat{Q}}(\hat{q}^*|\hat{\mathbf{s}}^*, \mathcal{D}) = \int_{\Theta} f_{\hat{Q}}(\hat{q}^*|\hat{\mathbf{s}}^*, \boldsymbol{\theta}) f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D}) \mathrm{d}\boldsymbol{\theta}, \tag{4.1}$$

where $f_{\hat{Q}}(\hat{q}^*|\hat{\mathbf{s}}^*, \boldsymbol{\theta})$ is obtained using the calibration function $h_{\boldsymbol{\theta}}(\hat{\mathbf{s}}^*)$ given the parameters $\boldsymbol{\theta}$.

For posterior inference, we place a prior distribution $f_{\boldsymbol{\theta}}(\boldsymbol{\theta})$ over the parameters $\boldsymbol{\theta}$ so that it is possible to obtain a posterior distribution $f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D})$ given the training data set $\mathcal{D}$. Thus, the posterior is defined by

$$f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D}) = f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathbf{s}, \hat{\mathbf{m}}) = \frac{f_{\hat{Q}}(\hat{\mathbf{m}}|\mathbf{s}, \boldsymbol{\theta}) f_{\boldsymbol{\theta}}(\boldsymbol{\theta})}{\int_{\Theta} f_{\hat{Q}}(\hat{\mathbf{m}}|\mathbf{s}, \boldsymbol{\theta}^*) f_{\boldsymbol{\theta}}(\boldsymbol{\theta}^*) \mathrm{d}\boldsymbol{\theta}^*}, \tag{4.2}$$

using $\hat{\mathbf{m}} = (\hat{m}_1, \ldots, \hat{m}_N)^\top$ and $\mathbf{s} = (\hat{\mathbf{s}}_1, \ldots, \hat{\mathbf{s}}_N)^\top$, where $f_{\hat{Q}}(\hat{\mathbf{m}}|\mathbf{s}, \boldsymbol{\theta})$ is the model likelihood of $h_{\boldsymbol{\theta}}$ given the data set $\mathcal{D}$. Thus, the posterior reflects the probability distribution for the model parameters $\boldsymbol{\theta}$ given the training data $\mathcal{D}$. However, since the integral over the complete parameter set Θ is intractable, it is not possible to analytically obtain the posterior for $\boldsymbol{\theta}$ [102, 68]. Therefore, we adapt SVI as an approximation method to infer the posterior [102, 68, 101] where a variational distribution $f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})$ of known functional form is used with distribution parameters $\boldsymbol{\omega}$ to approximate the real posterior, so that $f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega}) \approx f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D})$. Note that using variational distributions, possible correlations between the calibration parameters are neglected. Furthermore, using Markov-Chain Monte-Carlo (MCMC) for approximating the posterior is known to be asymptotically exact [103], whereas SVI methods are limited by the use of variational distributions. However, SVI comes with computationally low costs and scales well to large data sets compared to MCMC methods. Thus, we further use SVI for approximating the posterior using a Gaussian to implement the variational distribution as it has a known functional form and is easy to evaluate.

The distribution parameters $\boldsymbol{\omega}$ of $f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})$ are obtained using the Evidence Lower Bound (ELBO) loss [102, 101]. The basic idea is that the evidence $f(\mathcal{D})$ is the target distribution to maximize the model likelihood given a certain parameter set $\boldsymbol{\theta}$. The ELBO loss treats the evidence $f(\mathcal{D})$ as an upper bound during the maximization of the model likelihood and is derived by

$$\log\big(f(\mathcal{D})\big) = \log\left(\int_\Theta f(\mathcal{D},\boldsymbol{\theta})\mathrm{d}\boldsymbol{\theta}\right) \tag{4.3}$$

$$= \log\left(\int_\Theta f(\mathcal{D},\boldsymbol{\theta})\frac{f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})}{f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})}\mathrm{d}\boldsymbol{\theta}\right) \tag{4.4}$$

$$= \log\left(\mathbb{E}_{f^*_{\boldsymbol{\theta}}}\left[\frac{f_{\boldsymbol{\theta}}(\mathcal{D},\boldsymbol{\theta})}{f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})}\right]\right) \tag{4.5}$$

$$\geq \mathbb{E}_{f^*_{\boldsymbol{\theta}}}\Big[\log\big(f_{\boldsymbol{\theta}}(\mathcal{D},\boldsymbol{\theta})\big)\Big] - \mathbb{E}_{f^*_{\boldsymbol{\theta}}}\Big[\log\big(f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})\big)\Big], \tag{4.6}$$

using Jensen's inequality on the log probability of observations. Note that the ELBO is related to the Kullback-Leibler divergence between the variational distribution $f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})$ and the posterior $f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D})$ by

$$D_{\mathrm{KL}}\big(f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})||f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D})\big) = \mathbb{E}_{f^*_{\boldsymbol{\theta}}}\left[\log\left(\frac{f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})}{f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D})}\right)\right] \tag{4.7}$$

$$= \mathbb{E}_{f^*_{\boldsymbol{\theta}}}\Big[\log\big(f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})\big)\Big] - \mathbb{E}_{f^*_{\boldsymbol{\theta}}}\Big[\log\big(f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D})\big)\Big] \tag{4.8}$$

$$= \mathbb{E}_{f^*_{\boldsymbol{\theta}}}\Big[\log\big(f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})\big)\Big] - \mathbb{E}_{f^*_{\boldsymbol{\theta}}}\Big[\log\big(f_{\boldsymbol{\theta}}(\mathcal{D},\boldsymbol{\theta})\big)\Big] + \log\big(f(\mathcal{D})\big) \tag{4.9}$$

$$= -\Big[\mathbb{E}_{f^*_{\boldsymbol{\theta}}}\Big[\log\big(f_{\boldsymbol{\theta}}(\mathcal{D},\boldsymbol{\theta})\big)\Big] - \mathbb{E}_{f^*_{\boldsymbol{\theta}}}\Big[\log\big(f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})\big)\Big]\Big] + \log\big(f(\mathcal{D})\big), \tag{4.10}$$

which is the negative ELBO and the log evidence $f(\mathcal{D})$. Thus, maximizing the ELBO leads to a minimization of the Kullback-Leibler divergence in (4.7). Similar to the training of neural networks, SVI utilizes a gradient descent approach using backpropagation to optimize the variational distribution parameters $\boldsymbol{\omega}$. Once the approximate distribution for the posterior has been learned, we can plug in the variational distribution into the posterior predictive in (4.1) by

$$f_{\hat{Q}}(\hat{q}^*|\hat{\mathbf{s}}^*,\mathcal{D}) = \int_\Theta f_{\hat{Q}}(\hat{q}^*|\hat{\mathbf{s}}^*,\boldsymbol{\theta}) f_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\mathcal{D})\mathrm{d}\boldsymbol{\theta} \tag{4.11}$$

$$\approx \int_\Theta f_{\hat{Q}}(\hat{q}^*|\hat{\mathbf{s}}^*,\boldsymbol{\theta}) f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})\mathrm{d}\boldsymbol{\theta}, \tag{4.12}$$

which denotes the probability distribution of $\hat{Q}^*$ given the new sample $\hat{\mathbf{s}}^*$ and the training set $\mathcal{D}$. The distribution for $\hat{Q}^*$ should express the uncertainty for the calibrated probability. However, for subsequent applications such as object detection (cf. Chap. 6), it is mandatory to provide a scalar for the calibrated probability

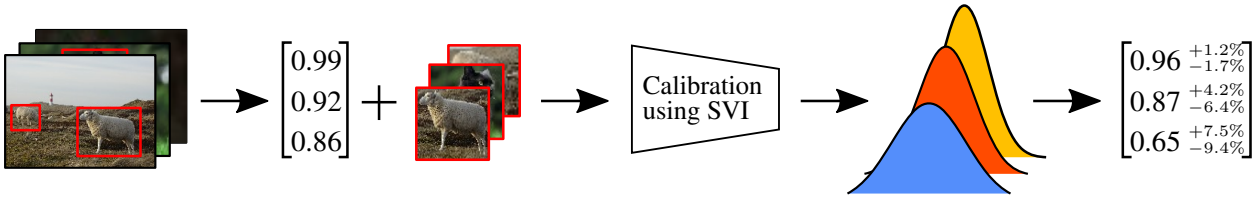


Figure 4.2: Concept of Bayesian confidence calibration [3, p. 2, Fig. 2]. A detector estimates several objects with a certain confidence and a certain position within an image. In Bayesian confidence calibration, the parameters of the calibration function are replaced by variational (Gaussian) distribution and learned by Stochastic Variational Inference (SVI). When the calibration method is applied to the detector output, we sample from the variational distributions to obtain multiple parameter sets. These parameter sets are then used to generate a sample distribution for the calibration output that represents the uncertainty within the recalibrated confidence.

to finally construct the Bernoulli $\hat{M}^*$. Thus, the expectation $\mathbb{E}_{\hat{Q}^*}[\hat{Q}^*|\hat{\mathbf{s}}^*, \mathcal{D}]$ can be used to assess a mean estimate for the calibrated confidence. In application, we sample T parameter combinations $\hat{\boldsymbol{\theta}} \in \Theta$ from the posterior distribution $\hat{\boldsymbol{\theta}} \sim f^*_{\boldsymbol{\theta}}(\boldsymbol{\theta}|\boldsymbol{\omega})$ and perform calibration with each parameter set to approximate the expectation and the variance. We denote the mean by $\mu_{\hat{Q}}$ that is an approximation of the expectation to obtain the calibrated confidence by

$$\mathbb{E}_{\hat{Q}^*}[\hat{Q}^*|\hat{\mathbf{s}}^*, \mathcal{D}] \approx \frac{1}{T}\sum_{t=1}^{T} f_{\hat{Q}}(\hat{q}^*|\hat{\mathbf{s}}^*, \hat{\boldsymbol{\theta}}_t) = \mu_{\hat{Q}^*}, \tag{4.13}$$

which is used to construct the Bernoulli distribution for $\hat{M}^*$, so that

$$f_{\hat{M}^*}(\hat{m}^*|\mu_{\hat{Q}^*}) = \mu_{\hat{Q}^*}^{\hat{m}^*}(1-\mu_{\hat{Q}^*})^{1-\hat{m}^*}, \tag{4.14}$$

yields the final calibrated confidence and probability distribution for $\hat{M}^*$. The concept of Bayesian confidence calibration is schematically shown in Fig. 4.2. Note that this kind of calibration does not necessarily lead to a monotonically increasing calibration mapping any more which might affect the baseline average precision. This influence is thus investigated in our experiments for Bayesian confidence calibration. In general, it is also possible to obtain a variance by sampling from the posterior in the same way. However, we face some challenges for the uncertainty quantification. On the one hand, the final probability for $\hat{M}^*$ is expressed in terms of a Bernoulli distribution. Furthermore, we do not have a direct ground-truth for $\hat{Q}^*$ available to evaluate the epistemic uncertainty. We mitigate this problem by using the same approach to quantify the ground-truth information for $\hat{Q}^*$ as within the Detection Expected Calibration Error (D-ECE) calculation. We apply a binning scheme over the uncalibrated confidence estimates for all samples within a data set $\mathcal{D}$ and measure the precision within each bin. This yields the ground-truth information $\bar{P} \in [0, 1]$ for all samples in $\mathcal{D}$ in each bin which allows for an evaluation of the calibrated uncertainty. Similarly to the standard Expected Calibration Error (ECE) or D-ECE calculation, using a binning scheme leads to a modeling error which, however, we tolerate in our experiments for practical reasons.

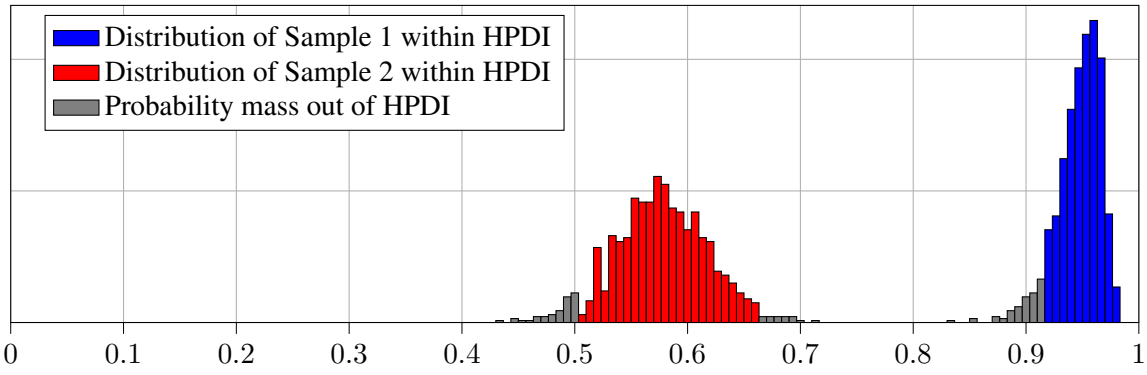


Figure 4.3: Qualitative example of two samples after Bayesian confidence calibration with their respective probability distributions for the calibrated confidence [3, p. 4, Fig. 3]. The probability distributions do not necessarily follow a Gaussian distribution. Therefore, we seek for the highest posterior density interval (HPDI) to express the epistemic uncertainty of a calibration mapping.

On the other hand, the probability distribution for the calibrated confidences might not necessarily follow a distribution of a known parametric form, e.g., a Gaussian. This is demonstrated in Fig. 4.3. Thus, a sampled variance might not properly reflect the epistemic uncertainty of a Bayesian confidence mapping. Therefore, we seek to express the epistemic calibration uncertainty in terms of a prediction interval. A prediction interval is represented by interval boundaries $a_\tau, b_\tau \in [0, 1]$ which enclose a certain probability mass $\tau \in [0, 1]$, so that

$$\int_{a_\tau}^{b_\tau} f_{\hat{Q}^*}(\hat{q}^*|\hat{\mathbf{s}}^*, \mathcal{D})\mathrm{d}\hat{q}^* = \tau. \tag{4.15}$$

We further denote the prediction interval as $\hat{C}_\tau^* = [\hat{a}_\tau^*, \hat{b}_\tau^*]$. The advantage of using prediction intervals is that these intervals reflect the probability for a new sample located within the interval boundaries. However, since prediction intervals are not uniquely defined, we consider the Highest Posterior Density Interval (HPDI) to express epistemic uncertainty, which reflects the narrowest credible interval given a certain confidence level τ. Using the HPDI, we can evaluate the consistency of the predicted uncertainty by comparing the prediction interval with the observed interval coverage. This is a requirement by the definition for uncertainty calibration [34] and will be discussed in Chap. 5 in more detail. For this reason, we adapt the Prediction Interval Coverage Probability (PICP) [104] as a metric that measures the fraction of samples whose ground-truth score fall into the estimated prediction interval for a certain confidence level, so that

$$\mathrm{PICP}(\tau) := \frac{1}{N}\sum_{n=1}^{N} \mathbb{1}(\bar{P}_n \in \hat{C}_{\tau,n}). \tag{4.16}$$

Furthermore, we use the Mean Prediction Interval Width (MPIW) [104] as a complementary metric to evaluate the sharpness of the predicted distribution. Thus, we finally considered all necessary aspects to determine and evaluate the epistemic uncertainty of a Bayesian calibration method.

4.2 Experiments for Bayesian Confidence Calibration

For the experiments of the Bayesian confidence calibration methods, we use the same setup as within our experiments for object detection evaluation in Sec. 3.4.1 and use a `Faster R-CNN`, `RetinaNet`, and `Mask R-CNN` on the MS COCO and Cityscapes data sets, respectively. We further use the D-ECE, Brier score, and Negative Log Likelihood (NLL) to evaluate the calibration performance of the Bayesian methods. Furthermore, we also use the PICP and MPIW [104] metrics for a fixed prediction interval of $\tau = 0.95$ to evaluate the uncertainty (for a detailed description of these metrics, see Sec. 4.1). For further uncertainty evaluation, we perform inference of each network on a different data set. For example, a `Faster R-CNN`, which has been trained for MS COCO, is also used for inference on the Cityscapes validation set. We reuse the respective calibration methods for each network configuration and do not perform a retraining of the methods on the new data sets. In this way, we are able to study the effect of a possible covariate shift on the uncertainty of the calibrated confidence. The evaluation results for the confidence-only case are shown in Tab. 4.1, as well as for the position-dependent calibration cases in Tab. 4.2 and Tab. 4.3 for conditional independent and conditional dependent calibration, respectively. Additionally, we compare the Bayesian calibration methods with their deterministic counterparts (cf. Tab. 3.1 and Tab. 3.1) that is shown in Fig. 4.4.

As already stated, Bayesian confidence calibration does not guarantee a monotonically increasing calibration mapping and thus might affect the baseline average precision. However, we only find a marginal effect of the Bayesian confidence calibration on the average precision which, thus, does not degenerate baseline performance. The Bayesian calibration methods (confidence-only and position-dependent) consistently reduce miscalibration and show a similar calibration performance compared to the standard methods which are trained by simple maximum likelihood estimation. This also holds for the miscalibration on the foreign data sets in most cases. Moreover, we observe a higher epistemic uncertainty when calibration is applied on the data sets for which the calibration methods have not been trained for (see MPIW scores). In our

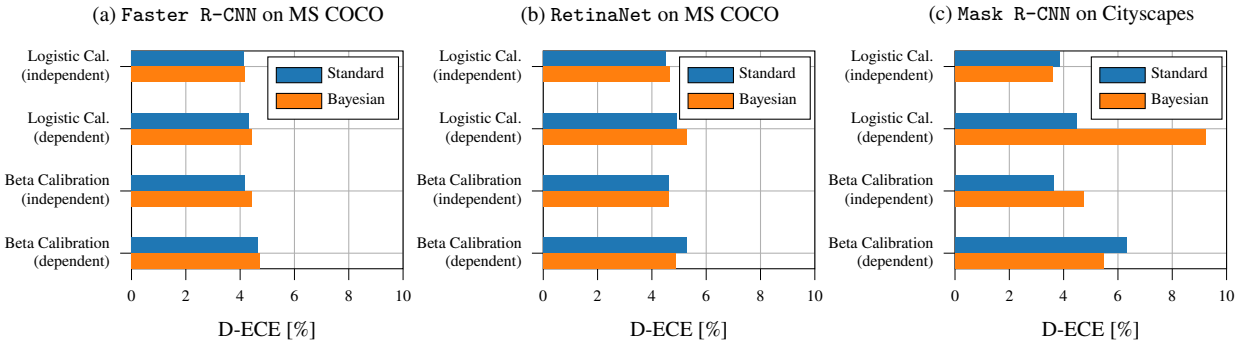


Figure 4.4: Comparison of the calibration performance between standard calibration and Bayesian confidence calibration for the different scaling methods and on different networks and data sets. Apart from one exception, we observe a consistent calibration performance of the Bayesian methods compared to their standard counterparts which are built using maximum likelihood estimation.

Table 4.1: Calibration results for Bayesian confidence calibration where only the predicted confidence $\hat{P}$ is used for calibration and evaluation. Moreover, the networks as well as the trained calibration methods are used on a different data set to evaluate the effect of a possible covariate shift on the Bayesian confidence calibration. The best scores are highlighted in bold. In this case, logistic and beta calibration show equal calibration performance. In addition, both calibration methods consistently indicate a higher uncertainty on the data sets for which they have not been trained for.

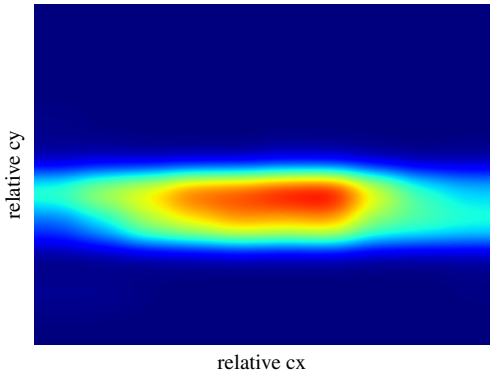
Network	IoU	Evaluation data set	Calibration method	D-ECE	Brier	NLL	AUPRC	PICP	MPIW
Faster R-CNN (trained on MS COCO)	0.50	MS COCO	Uncalibrated	0.153	0.176	0.536	0.920	-	-
			Logistic Cal.	**0.021**	**0.142**	**0.433**	0.920	**0.900**	0.085
			Beta Cal.	0.022	**0.142**	**0.433**	0.920	0.872	0.092
		Cityscapes	Uncalibrated	**0.093**	0.168	0.552	0.887	-	-
			Logistic Cal.	0.115	**0.166**	**0.505**	0.887	0.971	0.116
			Beta Cal.	0.117	0.167	0.509	0.887	**0.954**	0.113
	0.75	MS COCO	Uncalibrated	0.294	0.257	0.829	0.866	-	-
			Logistic Cal.	**0.027**	**0.144**	0.449	0.866	**0.922**	0.085
			Beta Cal.	**0.027**	**0.144**	**0.448**	0.866	**0.978**	0.093
		Cityscapes	Uncalibrated	0.296	0.275	0.917	0.814	-	-
			Logistic Cal.	**0.064**	0.163	0.499	0.814	**0.969**	0.121
			Beta Cal.	0.066	**0.162**	**0.498**	0.814	0.995	0.123
RetinaNet (trained on MS COCO)	0.50	MS COCO	Uncalibrated	0.083	0.157	0.478	0.907	-	-
			Logistic Cal.	**0.024**	**0.149**	**0.451**	0.907	**0.894**	0.076
			Beta Cal.	0.026	0.150	0.452	0.907	0.852	0.095
		Cityscapes	Uncalibrated	0.131	0.174	0.522	0.879	-	-
			Logistic Cal.	**0.115**	**0.171**	0.515	0.879	**0.941**	0.094
			Beta Cal.	0.116	**0.171**	**0.513**	0.879	0.887	0.116
	0.75	MS COCO	Uncalibrated	0.151	0.172	0.518	0.855	-	-
			Logistic Cal.	0.033	**0.140**	0.439	0.855	**0.950**	0.078
			Beta Cal.	**0.027**	**0.140**	**0.437**	0.855	0.985	0.095
		Cityscapes	Uncalibrated	0.149	0.190	0.565	0.789	-	-
			Logistic Cal.	0.063	0.161	0.495	0.789	**0.966**	0.097
			Beta Cal.	**0.058**	**0.160**	**0.492**	0.789	0.986	0.113
Mask R-CNN (trained on Cityscapes)	0.50	MS COCO	Uncalibrated	0.108	0.145	0.496	0.952	-	-
			Logistic Cal.	**0.030**	**0.124**	**0.378**	0.952	1.000	0.166
			Beta Cal.	0.042	0.126	0.382	0.952	0.783	0.098
		MS COCO	Uncalibrated	0.274	0.283	1.084	0.799	-	-
			Logistic Cal.	0.137	**0.215**	**0.641**	0.799	0.999	0.190
			Beta Cal.	**0.136**	0.216	0.683	0.799	**0.902**	0.132
	0.75	Cityscapes	Uncalibrated	0.296	0.269	1.055	0.896	-	-
			Logistic Cal.	**0.042**	**0.134**	**0.420**	0.896	0.997	0.203
			Beta Cal.	0.055	0.137	0.429	0.896	**0.958**	0.112
		MS COCO	Uncalibrated	0.439	0.397	1.632	0.689	-	-
			Logistic Cal.	**0.086**	**0.184**	**0.555**	0.689	1.000	0.205
			Beta Cal.	0.116	0.192	0.598	0.689	**0.986**	0.133

Table 4.2: Calibration results for Bayesian confidence calibration using the conditional **independent** scaling methods where all information $\hat{\mathbf{S}} = (\hat{P}, \hat{Y}, \hat{\mathbf{R}})^\top$ are used for calibration and evaluation. The underlined scores are the best calibration results compared to their conditional dependent counterparts in Tab. 4.3. For the position-dependent case, we observe an equal performance of both calibration methods. Although the calibration methods indicate a higher uncertainty on foreign data, the calibration uncertainty is consistently below the desired confidence level.

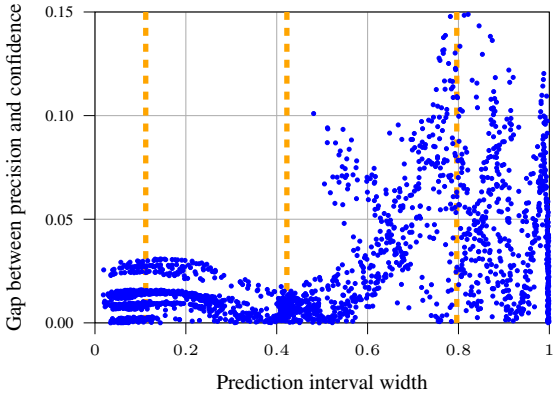
Network	IoU	Evaluation data set	Calibration method	D-ECE	Brier	NLL	AUPRC	PICP	MPIW
`Faster R-CNN` (trained on MS COCO)	0.50	MS COCO	Uncalibrated	0.119	0.176	0.536	**0.920**	-	-
			Logistic Cal.	**0.042**	**0.143**	**0.434**	**0.920**	0.627	0.108
			Beta Cal.	0.044	0.144	0.439	0.918	**0.738**	0.175
		Cityscapes	Uncalibrated	**0.086**	0.168	0.552	**0.887**	-	-
			Logistic Cal.	0.107	0.167	0.506	**0.887**	0.724	0.134
			Beta Cal.	0.090	**0.163**	**0.499**	0.886	**0.800**	0.231
	0.75	MS COCO	Uncalibrated	0.227	0.257	0.829	**0.866**	-	-
			Logistic Cal.	**0.045**	**0.145**	**0.450**	0.865	0.611	0.106
			Beta Cal.	0.048	0.146	0.453	0.863	**0.805**	0.163
		Cityscapes	Uncalibrated	0.274	0.275	0.917	0.814	-	-
			Logistic Cal.	**0.063**	**0.163**	**0.499**	0.814	0.748	0.151
			Beta Cal.	0.081	0.164	0.502	0.814	**0.881**	0.232
`RetinaNet` (trained on MS COCO)	0.50	MS COCO	Uncalibrated	0.072	0.157	0.478	0.907	-	-
			Logistic Cal.	**0.046**	**0.149**	**0.449**	**0.909**	0.612	0.112
			Beta Cal.	**0.046**	0.150	0.452	0.908	**0.710**	0.182
		Cityscapes	Uncalibrated	0.123	0.174	0.522	0.879	-	-
			Logistic Cal.	**0.109**	**0.172**	**0.519**	0.880	0.657	0.122
			Beta Cal.	0.114	0.179	0.546	**0.881**	**0.792**	0.252
	0.75	MS COCO	Uncalibrated	0.110	0.172	0.518	0.855	-	-
			Logistic Cal.	**0.046**	**0.139**	**0.437**	**0.856**	0.660	0.101
			Beta Cal.	**0.046**	**0.139**	0.439	0.855	**0.810**	0.165
		Cityscapes	Uncalibrated	0.136	0.190	0.565	0.789	-	-
			Logistic Cal.	**0.067**	**0.161**	**0.498**	0.790	0.703	0.121
			Beta Cal.	0.086	0.164	0.509	**0.791**	**0.870**	0.232
`Mask R-CNN` (trained on Cityscapes)	0.50	Cityscapes	Uncalibrated	0.102	0.145	0.496	0.952	-	-
			Logistic Cal.	**0.036**	**0.124**	**0.378**	0.952	**0.910**	0.185
			Beta Cal.	0.047	**0.124**	0.380	0.952	0.766	0.265
		MS COCO	Uncalibrated	0.234	0.283	1.084	**0.799**	-	-
			Logistic Cal.	**0.142**	**0.218**	**0.651**	0.784	**0.926**	0.278
			Beta Cal.	0.162	0.243	0.767	0.760	0.812	0.354
	0.75	Cityscapes	Uncalibrated	0.281	0.269	1.055	0.896	-	-
			Logistic Cal.	**0.056**	0.136	**0.423**	0.898	**0.969**	0.227
			Beta Cal.	0.061	**0.135**	0.427	**0.899**	0.881	0.253
		MS COCO	Uncalibrated	0.361	0.397	1.632	0.689	-	-
			Logistic Cal.	**0.118**	**0.190**	**0.572**	**0.703**	**0.983**	0.299
			Beta Cal.	0.134	0.209	0.658	0.666	0.888	0.336

Table 4.3: Calibration results for Bayesian confidence calibration using the conditional **dependent** calibration methods where all information $\hat{\mathbf{S}} = (\hat{P}, \hat{Y}, \hat{\mathbf{R}})^\top$ are used for calibration and evaluation. The underlined scores are the best calibration results of compared to their conditional independent counterparts in Tab. 4.2. The extended methods show similar calibration performance and further provide a meaningful uncertainty with a prediction interval coverage probability which is consistently close to the desired confidence level.

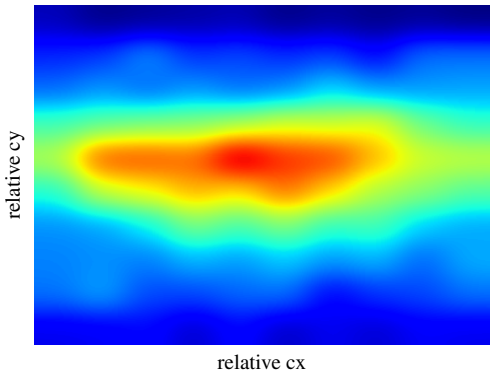
Network	IoU	Evaluation data set	Calibration method	D-ECE	Brier	NLL	AUPRC	PICP	MPIW
`Faster R-CNN` (trained on MS COCO)	0.50	MS COCO	Uncalibrated	0.119	0.176	0.536	**0.920**	-	-
			Logistic Cal.	**0.044**	**0.144**	**0.439**	0.919	0 **0.885**	0.284
			Beta Cal.	0.047	0.145	0.445	0.916	0.837	0.263
		Cityscapes	Uncalibrated	**0.086**	0.168	0.552	**0.887**	-	-
			Logistic Cal.	0.117	0.170	0.528	0.885	0.927	0.400
			Beta Cal.	0.105	**0.167**	**0.519**	0.885	**0.928**	0.369
	0.75	MS COCO	Uncalibrated	0.227	0.257	0.829	**0.866**	-	-
			Logistic Cal.	**0.047**	**0.146**	**0.457**	0.862	0.990	0.300
			Beta Cal.	**0.047**	0.147	0.458	0.862	**0.942**	0.262
		Cityscapes	Uncalibrated	0.274	0.275	0.917	**0.814**	-	-
			Logistic Cal.	0.077	**0.165**	0.508	**0.814**	0.991	0.464
			Beta Cal.	**0.068**	**0.165**	**0.507**	0.806	**0.986**	0.415
`RetinaNet` (trained on MS COCO)	0.50	MS COCO	Uncalibrated	0.072	0.157	0.478	0.907	-	-
			Logistic Cal.	0.053	**0.151**	**0.456**	**0.908**	**0.878**	0.333
			Beta Cal.	**0.049**	**0.151**	**0.456**	0.906	0.802	0.292
		Cityscapes	Uncalibrated	0.123	0.174	0.522	0.879	-	-
			Logistic Cal.	0.118	0.171	0.527	0.877	**0.962**	0.463
			Beta Cal.	**0.088**	**0.167**	**0.516**	**0.881**	0.824	0.336
	0.75	MS COCO	Uncalibrated	0.110	0.172	0.518	0.855	-	-
			Logistic Cal.	0.048	**0.140**	**0.439**	**0.856**	**0.978**	0.318
			Beta Cal.	**0.047**	0.141	0.441	0.853	0.906	0.265
		Cityscapes	Uncalibrated	0.136	0.190	0.565	0.789	-	-
			Logistic Cal.	**0.066**	**0.161**	**0.497**	0.784	0.995	0.477
			Beta Cal.	0.080	0.163	0.506	**0.792**	**0.968**	0.361
`Mask R-CNN` (trained on Cityscapes)	0.50	Cityscapes	Uncalibrated	0.102	0.145	0.496	**0.952**	-	-
			Logistic Cal.	0.092	0.144	0.472	0.935	0.655	0.365
			Beta Cal.	**0.055**	**0.131**	**0.395**	0.949	**0.776**	0.475
		MS COCO	Uncalibrated	0.234	0.283	1.084	**0.799**	-	-
			Logistic Cal.	**0.155**	**0.241**	0.815	0.751	0.862	0.476
			Beta Cal.	0.177	0.242	**0.743**	0.773	**0.885**	0.588
	0.75	Cityscapes	Uncalibrated	0.281	0.269	1.055	**0.896**	-	-
			Logistic Cal.	0.081	0.150	0.480	0.889	0.809	0.574
			Beta Cal.	**0.067**	**0.140**	**0.436**	0.895	**0.945**	0.536
		MS COCO	Uncalibrated	0.361	0.397	1.632	0.689	-	-
			Logistic Cal.	**0.101**	**0.193**	**0.598**	**0.685**	**0.947**	0.665
			Beta Cal.	0.140	0.213	0.648	**0.675**	0.969	0.556



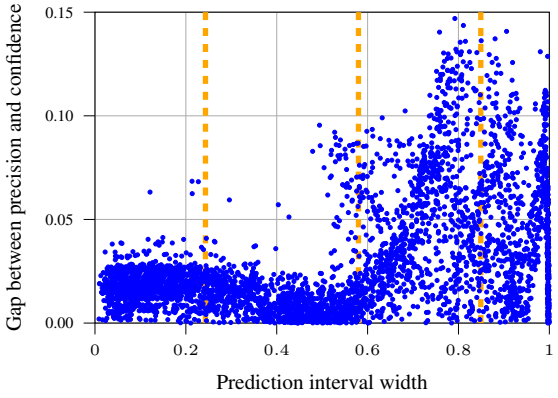
(i) Data distribution for the c_x and c_y position of objects for Cityscapes.



(ii) Calibration error w.r.t. the epistemic uncertainty of single predictions for Cityscapes.



(iii) Data distribution for the c_x and c_y position of objects for MS COCO.



(iv) Calibration error w.r.t. the epistemic uncertainty of single predictions for MS COCO.

Figure 4.5: Data distribution of the c_x and c_y position and the correlation between epistemic calibration uncertainty and calibration error of a logistic calibration model for a `Mask R-CNN` on different data sets. [3, p. 5, Fig. 4]. The `Mask R-CNN` as well as the confidence calibration have been trained on the Cityscapes data set. In the top row (b), we evaluate the correlation between epistemic uncertainty obtained by Bayesian calibration and the calibration error itself for the Cityscapes data set. The orange-dotted lines denote the $\{25, 50, 75\}$ percentiles of the samples. We repeat this for the MS COCO data set (d) to evaluate a possible change in uncertainty. We can inspect that the calibration error increases for a large epistemic uncertainty in both cases. In addition, the average epistemic uncertainty increases for the MS COCO data set as indicated by the percentiles.

experiments, the confidence-only methods provide consistent uncertainty quantifications whose prediction interval coverage probability (PICP) is close to the desired confidence level of $\tau = 0.95$. We further inspect the position-dependent calibration and compare the conditionally independent methods with their conditionally dependent counterparts. Although the conditionally independent calibration methods provide a better

calibration mapping in general, they underestimate the epistemic uncertainty in most cases. In contrast, the conditionally dependent calibration methods consistently provide good uncertainty estimates which are close to the desired confidence level. This also holds for calibration on the foreign data sets. For further uncertainty evaluation, a visual example is given in Fig. 4.5 which demonstrates the effect of Bayesian confidence calibration on foreign data sets. In this example, the Bayesian calibration method was trained on the Cityscapes data set where most objects are located in the image center. If we compare the epistemic uncertainty with the miscalibration for each sample individually, we can observe an increasing miscalibration for increasing epistemic uncertainty. This is the desired effect for our Bayesian confidence calibration methods. In the next step, this trained calibration mapping is used for calibration on the MS COCO data set. We observe a wider distribution for the location of objects within the data set compared to Cityscapes. On the one hand, we also observe an increasing miscalibration for increasing epistemic uncertainty. On the other hand, the epistemic uncertainty is higher on average compared to the calibration results for the Cityscapes data set. This is visualized by the orange-dotted lines which indicate the $\{25, 50, 75\}$ percentiles of the samples. In both scenarios, we observe an increasing calibration error for samples with a prediction interval width above 0.5. This indicates that a reliable prediction for the calibrated confidence of such samples is a challenging task. One way to handle this phenomenon is to place a threshold for the prediction interval width to detect critical samples during the inference. The high variation of the calibration error for uncertain samples might also be reduced by using more training data. Therefore, we conclude that the epistemic uncertainty is a valuable indicator for a possibly higher miscalibration as well as a sufficient criterion for a possible covariate shift of the data during inference.

4.3 Conclusion for Bayesian Confidence Calibration

A reliable uncertainty assessment is crucial especially for safety-critical applications. A step towards more safety is the usage of appropriate calibration methods to get a more interpretable assessment for the probability of being correct. However, confidence calibration might also be misleading in situations that are unknown either to the baseline object detector or to the calibration mapping itself. Thus, we are interested in the epistemic model uncertainty of such a calibration mapping to be able to indicate a possibly high model uncertainty in critical situations. For this reason, we derive the term of Bayesian confidence calibration that transfers the idea of epistemic uncertainty modeling in Bayesian statistics to the task of confidence calibration. In this context, we interpret the calibrated confidence as a random variable whose probability distribution is determined by probabilistic calibration methods. A prior distribution is placed over the calibration weights which allows for a probabilistic computation of the calibrated confidences. In this way, it is possible to quantify the (epistemic) model uncertainty of a calibration mapping itself which might be used to assess the reliability in calibrated confidence estimates.

The investigations for the Bayesian confidence calibration models show that it is possible to yield a qualitatively good calibration mapping by using stochastic variational inference for calibration training. On the one hand, the Bayesian calibration methods provide a meaningful uncertainty quantification for the epis-

temic uncertainty inherent in the calibration mapping itself. Especially the conditionally dependent calibration methods provide good uncertainty estimates for the epistemic calibration uncertainty. On the other hand, we find a connection between epistemic uncertainty and miscalibration which might be used as a sufficient criterion to detect a possible covariate shift within the calibration mapping. This allows for an indication of possibly unknown out-of-distribution samples during inference. However, for samples with a higher epistemic uncertainty, we also observe an increasing calibration error above a certain threshold. Thus, a reliable estimation of the confidence is a challenging task for these samples which needs further investigations. Nevertheless, we conclude that Bayesian confidence calibration is a valuable contribution for safety-critical applications as it allows for an additional indication of epistemic uncertainty without losing calibration performance. Finally, it might be used as an additional component for a possible out-of-distribution recognition. The assessment of the epistemic uncertainty for the object existence might also be used within an object tracking framework using a particle filter for the object confidence estimation. This is subject of future work.

5 Spatial Uncertainty Calibration

For environment perception, it is necessary to identify not only the semantic label of an object but also its position. Thus, in the scope of object detection, a neural network needs to infer the class as well as the position and shape of individual objects within an image. In this chapter, we focus on the regression branch of an object detection model. While the classification head commonly outputs a score indicating the model's belief about the predicted class (cf. Chap. 3), the regression branch is usually designed to output the position/shape information without any uncertainty information. However, a detection model can also be trained so that it outputs probabilistic estimates for the position information using Gaussian distributions [40, 71, 70, 16]. This has already been described in Sec. 2.2.2. On the one hand, this additional uncertainty can be interpreted as a self-assessment of the model's belief about the predicted position/shape information. On the other hand, the uncertainty can also be used for subsequent processes such as Kalman filtering for object tracking. Similar to semantic confidence calibration, the predicted variance can be interpreted as an estimate of the aleatoric uncertainty which is inherent in the model input [39]. We can evaluate if the predicted uncertainty matches the observed prediction error. This is of major importance especially if such a detection model is used in the context safety-critical applications such as autonomous driving. Similar to the methods for semantic confidence calibration, it is possible to examine the probabilistic forecaster regarding a systemic bias in its uncertainty estimations for the bounding box position. If we detect such a deviation, we can apply post-hoc calibration methods to perform a recalibration of the spatial uncertainty. The concept of uncertainty calibration is qualitatively shown in Fig. 5.1.

(a) Uncalibrated (b) GP-Beta (c) GP-Normal (mv.)

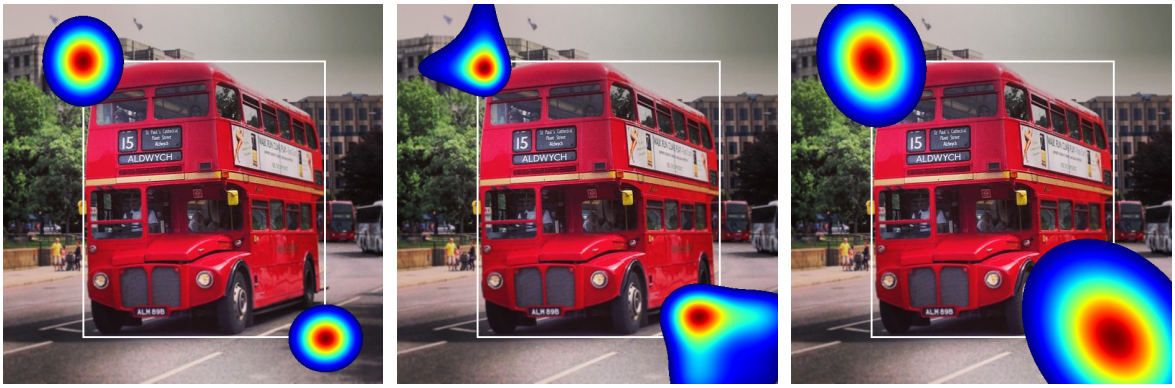


Figure 5.1: Qualitative example for spatial uncertainty calibration on predictions of a probabilistic `RetinaNet` (cf. Sec. 6.6) on the MS COCO data set [6, p. 2, Fig. 1]. (a) The probabilistic detection model outputs normal distribution for the position, width, and height information that are modeled as independent Gaussians. (b) The GP-Beta [41] is a non-parametric calibration method that is able to estimate a calibrated probability distribution of arbitrary shape. (c) In contrast, our multivariate GP-Normal (cf. Sec. 5.4.2) is a parametric method yielding a multivariate Gaussian as calibration output. Thus, this method is able to represent possible correlations between the dimensions.

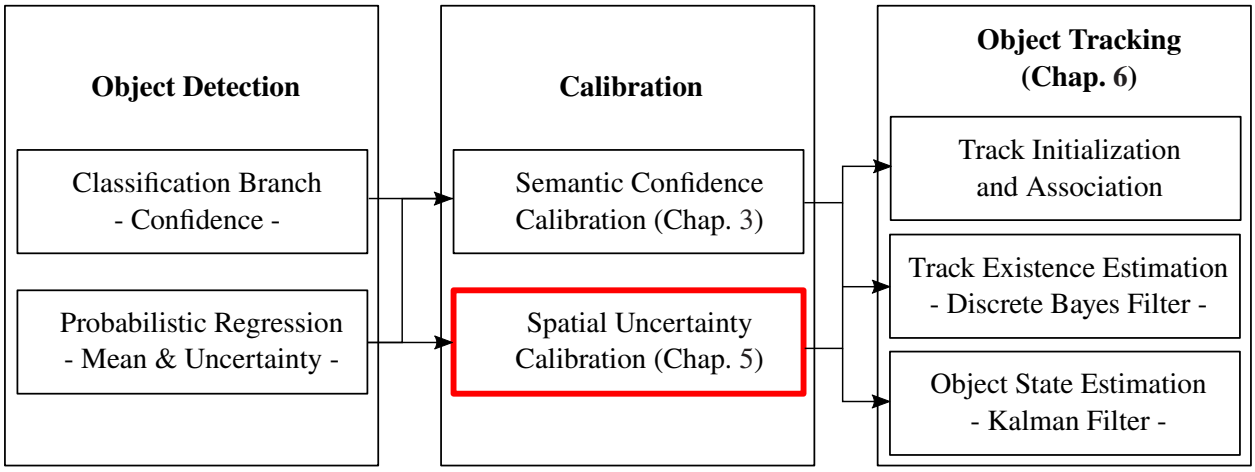


Figure 5.2: In this chapter, we focus on the evaluation and calibration of spatial uncertainty which is a crucial part of the image-based environment perception process. The recalibrated spatial uncertainty might be used for subsequent applications, e.g., object tracking (cf. Chap. 6). Therefore, a reliable assessment of spatial uncertainty is mandatory.

Reliable uncertainty evaluation is crucial especially if subsequent processes depend on these estimates, e.g., object tracking. Thus, we interpret the assessment as well as the recalibration of spatial uncertainty as part of a superordinate process chain that could be used e.g. for image-based environment perception in a vehicle. The concept of uncertainty calibration as a part of this process chain is schematically shown in Fig. 5.2. We start by reviewing the state-of-the-art for regression uncertainty calibration in Sec. 5.1. Furthermore, we review the existing definitions for regression uncertainty calibration and set them into a common mathematical context to each other. On the basis of these definitions, we describe the respective calibration metrics and derive new metrics to measure multivariate miscalibration in Sec. 5.2. In addition to these metrics, recent work has developed several post-hoc methods for the calibration of regression uncertainty. These techniques can be distinguished into parametric and non-parametric calibration methods. We review these methods and provide detailed mathematical descriptions in Sec. 5.3. Furthermore, we extend these methods in Sec. 5.4 to construct a new calibration method GP-Normal for a flexible and parametric recalibration scheme. These methods are evaluated on different data sets and for different object detection models in Sec. 5.5. Finally, we give a conclusion about our findings in Sec. 5.6.

Contributions: In summary, the following contributions can be found in this section::

- Common mathematical context for the definitions of regression uncertainty calibration.
- Derivation of the M-QCE and C-QCE metrics to measure multivariate regression calibration.
- New methods GP-Normal and GP-Cauchy for parametric recalibration using Gaussian processes.
- Joint multivariate calibration of multiple dimensions.
- Covariance estimation & recalibration for multivariate regression tasks.
- Extensive studies on the effect of uncertainty recalibration for probabilistic object detectors.

5.1 Related Work in the Context of Spatial Uncertainty Calibration

In the scope of regression uncertainty, recent work has proposed several definitions to define the term of calibration. The authors in [34] define the term of quantile calibration which requires that the predicted quantiles for a certain quantile level $\tau \in [0, 1]$ should cover $100\tau\%$ of the ground-truth scores given a finite data set. Furthermore, the authors adapt the Isotonic Regression calibration method known from semantic confidence calibration [85] and use this method to rescale the predicted cumulative distribution to achieve quantile calibration [34]. Thus, the calibration is applied in a post-hoc step after model training. In contrast, the authors in [41] argue that the definition of quantile calibration only faces the marginal coverage probability which does not consider the actually predicted probability distribution. The authors argue that this is in contrast to the common understanding of uncertainty calibration known from semantic confidence calibration which is conditioned on the actually predicted confidence [41] (cf. definition (3.1) in Sec. 3.2.1). Therefore, the authors propose the term of distribution calibration which requires that a predicted distribution should match the observed error distribution *given a certain probability distribution*. As calibration method, the authors further propose the GP-Beta method [41] which adapts the Beta calibration method from semantic confidence calibration [43] to perform a rescaling of the predicted cumulative distribution (similar to Isotonic Regression). To achieve distribution calibration, the authors adapt a Gaussian process to obtain the recalibration parameters for each distribution, individually. Independently, the authors in [42] and [18] propose the term of variance calibration which is designed for parametric normal distributions and requires that the predicted variance should match the observed mean squared error (which is equivalent to the observed variance) *for a certain variance level*. For variance calibration, the authors adapt the simple Temperature Scaling from semantic confidence calibration [13] to rescale the predicted variance by a single scalar [42, 18]. We use these definitions as well as the respective calibration methods for our examinations and set them in a common mathematical context in Sec. 5.2 and Sec. 5.3, respectively.

Recent work has developed a parametric approach to construct a probabilistic object detection model [40, 71, 70, 16]. A different approach of yielding aleatoric uncertainty is quantile regression [105, 106] where a model does not predict a certain score but directly infers the quantile boundaries. However, quantile regression has not been used for object detection so far but is definitely an interesting approach to investigate for future work. Besides the previously mentioned post-hoc calibration methods, recent work has also proposed techniques to achieve intrinsically calibrated probabilistic models during model training. The authors in [15] propose a calibration loss which adds a second regularization term that aims to minimize the difference between predicted variance and observed squared error. Another approach provided by [107] adapts maximum mean discrepancy (MMD) to perform a distribution matching between predicted and observed distribution during model training. Similarly, the authors in [108] propose the method f-Cal which is also used for a distribution matching during model training. The advantage of these calibration techniques is that they do not require a dedicated held-out calibration training set. However, as already mentioned in Sec. 3.1, the drawback of calibration during model training is that they aim to calibrate against the model's performance on the training set which is commonly much better compared to unseen data during inference. This may lead to a distortion after calibration. Therefore, we focus on post-hoc methods for regression calibration.

5.2 Definitions and Metrics for Uncertainty Calibration

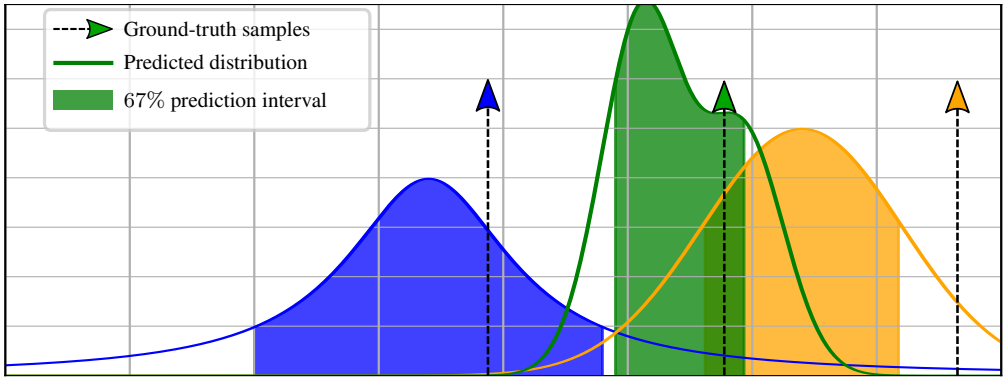
In this section, we review the definitions of quantile calibration [34], distribution calibration [41], and variance calibration [42, 18] for regression uncertainty calibration and place them in a common mathematical context. For this purpose, we follow the mathematical notation of the previous Chap. 3 and consider an object detection model that predicts individual objects with a certain label $\hat{Y} \in \mathcal{Y} = \{1, ..., K\}$ with an according confidence score $\hat{P} \in [0, 1]$ given the input samples $\mathbf{X} \in \mathcal{X}$, where K denotes the available number of classes. These predictions aim to target the real objects with ground-truth information for the class $\bar{Y} \in \mathcal{Y}$. Furthermore, let $\mathcal{R}$ denote the set of all possible bounding boxes with L as the size of the used box encoding. Commonly an encoding with the position, width, and height is used.

In contrast to the label prediction, an object detection model does not assess the uncertainty of the predicted object location by default. Therefore, we consider a probabilistic object detector [40, 71, 16] (cf. Sec. 2.2.2) that interprets the regression output $\hat{\mathbf{R}}$ as a Gaussian so that $\hat{\mathbf{R}}|\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}}, \boldsymbol{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}})$ with mean vector $\boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}} \in \mathcal{R}$ and the variances $\sigma_1^2, \ldots, \sigma_L^2 \in \mathbb{R}_{>0}^L$ for each bounding box quantity, so that $\boldsymbol{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}} = \text{diag}(\sigma_1^2, \ldots, \sigma_L^2)$. Thus, the network output for the bounding box regression can be interpreted as a random variable with a Probability Density Function (PDF) denoted by $f_{\hat{\mathbf{R}}|\mathbf{X}}(\hat{\mathbf{r}}) = \mathcal{N}(\hat{\mathbf{r}}; \boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}}, \boldsymbol{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}})$ which targets the true object location $\bar{\mathbf{R}} \in \mathcal{R}$. Let further denote $F_{\hat{\mathbf{R}}|\mathbf{X}}(\hat{\mathbf{r}})$ as the respective Cumulative Density Function (CDF) where $F : \mathcal{R} \rightarrow [0, 1]$. In the following, we also need the quantile function that returns the quantile boundaries for each bounding box quantity given a certain quantile $\tau \in [0, 1]$. However, the quantile function is only defined for the univariate case. Given the univariate CDF $F_{\hat{R}|\mathbf{X}}(\hat{\mathbf{r}})$ for a single bounding box quantity $\hat{R}$, the respective quantile function is denoted by $F_{\hat{R}|\mathbf{X}}^{-1}(\tau)$ so that $F_{\hat{\mathbf{R}}|\mathbf{X}}^{-1} : [0, 1] \rightarrow \mathbb{R}$. Note that using independent quantile functions for each bounding box quantity leads to a negligence of possible correlations.

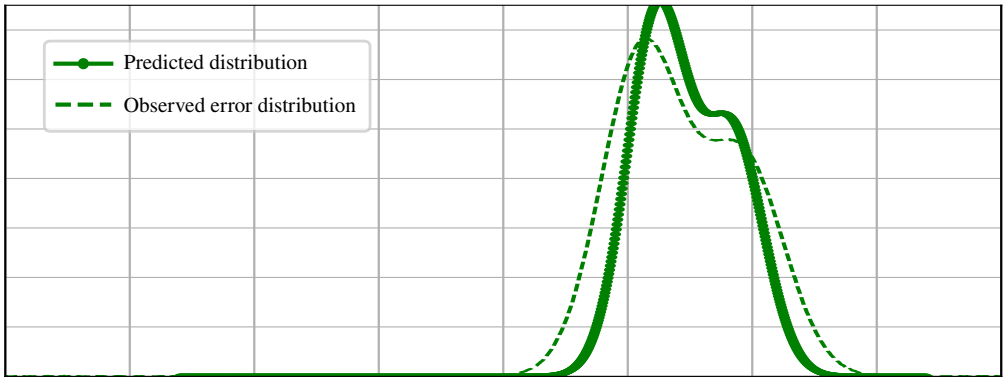
The target of probabilistic object detection is to predict the mean as precisely as possible to the ground-truth objects, but also to indicate a high uncertainty if the model fails to reliably predict the object position or shape. Thus, it is required to evaluate the quality of the predicted uncertainties and to examine for calibration. In the following, we review the different definitions of uncertainty calibration for the task of regression. We give an overview over the different types of regression calibration in Fig. 5.3. Note that the probabilistic object detector does not predict any covariances between the bounding box quantities which yields multiple independent normal distributions for each quantity. Since most of the existing definitions for regression calibration face only univariate probability distributions, we further revisit the definitions for single bounding box quantities which are denoted by $\hat{R} \sim \mathcal{N}(\mu_{\hat{R}|\mathbf{X}}, \sigma_{\hat{R}|\mathbf{X}}^2)$ for notational simplicity.

5.2.1 Quantile Calibration

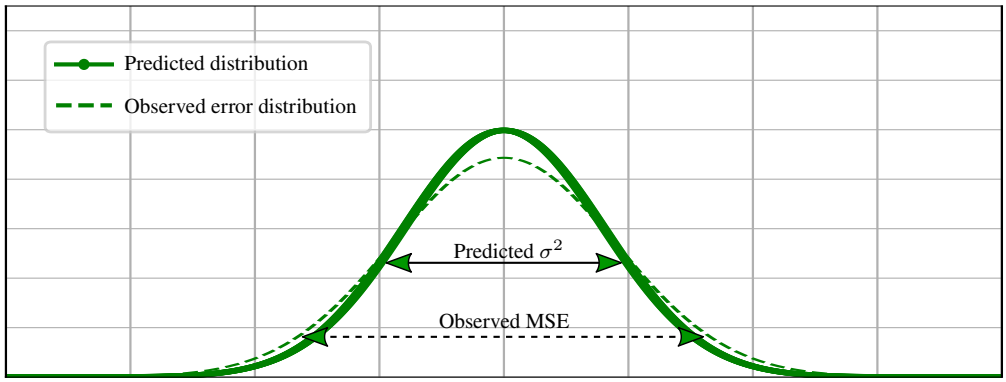
The first definition for uncertainty calibration is quantile calibration that has initially been proposed by the authors in [34]. A probabilistic forecaster is quantile-calibrated if the predicted quantiles for a quantile level τ cover $100\tau\%$ of the ground-truth samples. For example, consider 100 samples where a probabilistic



(i) Principle of quantile calibration [34] demonstrated by 3 predicted distributions with a prediction interval of 67%. The goal within quantile calibration is to cover approx. 67% of the ground-truth samples within the predicted quantiles.



(ii) Principle of distribution calibration [41]. The definition for distribution calibration requires that the observed error distribution matches the predicted distribution given all probability distributions with the same shape, e.g., as shown above.



(iii) Principle of variance calibration [42, 18]. For all predicted normal distributions with the same variance, the term of variance calibration requires that the observed variance, i.e., the observed Mean Squared Error (MSE) matches the predicted variance.

Figure 5.3: Overview over the different definitions for regression uncertainty calibration. We distinguish between (a) quantile calibration, (b) distribution calibration, and (c) variance calibration.

forecaster predicts a mean and variance that targets the ground-truth value. If we consider a quantile level of 0.8, we would expect that 80 of the ground-truth scores are covered by the predicted intervals. This must hold for all quantile levels $\tau \in [0, 1]$. Therefore, a probabilistic object detector is quantile-calibrated if

$$\mathbb{P}\big(\bar{R} \leq F^{-1}_{\hat{R}|\mathbf{X}}(\tau)\big) = \tau, \quad \forall \tau \in [0, 1], \tag{5.1}$$

holds for each bounding box dimension [34]. This definition also holds for two sided quantiles $\tau = \tau_2 - \tau_1$ where $\tau_1 < \tau_2$ [34], so that

$$\mathbb{P}\big(F^{-1}_{\hat{R}|\mathbf{X}}(\tau_1) \leq \bar{R} \leq F^{-1}_{\hat{R}|\mathbf{X}}(\tau_2)\big) = \tau_2 - \tau_1, \quad \forall \tau_1, \tau_2 \in [0, 1]. \tag{5.2}$$

The principle of quantile calibration is schematically shown in Fig. 5.3i. A probabilistic forecaster is commonly evaluated for quantile calibration using the Pinball loss $\mathcal{L}_{\text{Pin}}$ [109] for a certain τ that is defined by

$$\mathcal{L}_{\text{Pin}}(\tau) := \begin{cases} \big(\bar{R} - F^{-1}_{\hat{R}|\mathbf{X}}(\tau)\big)\tau & \text{if } \bar{R} \geq F^{-1}_{\hat{R}|\mathbf{X}}(\tau) \\ \big(F^{-1}_{\hat{R}|\mathbf{X}}(\tau) - \bar{R}\big)(1 - \tau) & \text{if } \bar{R} < F^{-1}_{\hat{R}|\mathbf{X}}(\tau) \end{cases}. \tag{5.3}$$

A mean $\overline{\mathcal{L}}_{\text{Pin}} := \mathbb{E}_\tau[\mathcal{L}_{\text{Pin}}(\tau)]$ can also be used to denote the calibration properties for several quantile levels. Further metrics are the Prediction Interval Coverage Probability (PICP) (cf. (4.16) in Sec. 4.1) [104] and the Mean Prediction Interval Width (MPIW) [104] for a certain τ (both have been introduced in Sec. 4.1).

A major difficulty for measuring quantile calibration is that prediction intervals for certain quantile levels are not uniquely defined. Therefore, we seek for the Highest Posterior Density Interval (HPDI) which denotes the narrowest prediction interval given a certain quantile level τ. The computation of the HPDI for non-parametric probability distributions requires a numerical approach for approximation. In contrast, it is considerably easier to determine the HPDI and thus the prediction interval coverage of a Gaussian distribution with a known parametric form. In addition, we can also determine the Highest Posterior Density Region (HPDR) which is the multivariate counterpart of the HPDI and reflects a certain region with probability mass τ for a certain quantile τ. Similar to the univariate case, we would expect that approx. $100\tau\%$ of the ground-truth samples are covered by this prediction region. To determine the prediction region coverage of the true value vector $\bar{\mathbf{R}}$ given a predicted Gaussian distribution with mean $\boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}}$ and covariance matrix $\boldsymbol{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}}$, we adapt the Normalized Estimation Error Squared (NEES) from Kalman filter consistency evaluation [46, pp. 232] [47, pp. 292] that is defined by

$$\epsilon_{\hat{\mathbf{R}}|\mathbf{X}} := \big(\bar{\mathbf{R}} - \boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}}\big)^\top \boldsymbol{\Sigma}^{-1}_{\hat{\mathbf{R}}|\mathbf{X}} \big(\bar{\mathbf{R}} - \boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}}\big), \tag{5.4}$$

which is also known as the squared Mahalanobis distance between the predicted distribution and the ground-truth vector [110]. Furthermore, we assume that the estimator has no bias in its predictions. Thus, the prediction interval coverage for the ground-truth vector $\bar{\mathbf{R}}$ can be determined using the χ^2-test. It is well known, see e.g. [47], that the NEES can be interpreted as the sum of L independent squared random variables with zero mean and unit variance [47, p. 295]. This sum is also represented by a χ^2_L distribution

with L degrees of freedom [47, p. 295]. The NEES $\epsilon_{\hat{\mathbf{R}}|\mathbf{X}}$ is highly connected to the HPDR for Gaussian distributions as its equation in (5.4) can also be interpreted as an equation for an ellipsoid which describes the contours/isolines for certain quantile levels. To test for prediction interval coverage, it is only required to determine if the radius of the ellipsoid for the ground-truth vector $\bar{\mathbf{R}}$ is below the radius for the target quantile which is given by $\chi^2_L(\tau)$[1]. Therefore, for a certain sample $\mathbf{x} \in \mathcal{X}$, the ground-truth vector is covered by the estimated prediction interval with a certain quantile level τ, if $\epsilon_{\hat{\mathbf{R}}|\mathbf{x}} \leq \chi^2_L(\tau)$ is fulfilled. More formally, we can denote the gap between the desired quantile level τ and the observed quantile coverage by

$$\left| \mathbb{P}\big(\epsilon_{\hat{\mathbf{R}}|\mathbf{X}} \leq \chi^2_L(\tau)\big) - \tau \right|. \tag{5.5}$$

We use this derivation to construct the Marginal Quantile Calibration Error (M-QCE) that is a new metric to measure for quantile calibration given multivariate normal distributions. The advantage of the M-QCE is that it tests for quantile calibration given multivariate normal distributions that might also represent possible correlations between the random variables. The M-QCE is related to the PICP but determines the absolute difference between prediction interval coverage probability and the target quantile. Therefore, the M-QCE can directly be interpreted as an error metric reflecting the deviation between expected and observed quantile coverage. On a finite data set $\mathcal{D} = \big\{(\bar{\mathbf{r}}_n, \boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{x}_n}, \boldsymbol{\Sigma}_{\hat{\mathbf{R}}|\mathbf{x}_n})\big\}_{n=1}^{N}$ with N samples, the M-QCE is designed to approximate (5.5) by

$$\text{M-QCE}(\tau) := \left| \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\big(\epsilon_{\hat{\mathbf{R}}|\mathbf{x}_n} \leq \chi^2_L(\tau)\big) - \tau \right|. \tag{5.6}$$

It is also possible to denote a mean $\overline{\text{M-QCE}} := \mathbb{E}_\tau[\text{M-QCE}(\tau)]$ for several quantile levels to measure the overall properties for quantile calibration of a probabilistic forecaster.

5.2.2 Distribution Calibration

If we review the definition for semantic confidence calibration (cf. (3.1) in Sec. 3.2.1), we can see that the probability for the observed accuracy is conditioned on the predicted confidence. This ensures that a forecaster not only predicts globally (marginally) calibrated confidences but also must provide informative confidence estimates for each confidence level separately. In contrast, the definition for quantile calibration only considers the marginal probability over all predicted probability distributions. This allows a forecaster to have putatively good calibration properties if the predicted quantiles match the observed quantile coverage on average, even if it is poorly calibrated at a local level (e.g., for subsets with neighboring samples). Therefore, the authors in [41] recently introduced the definition of distribution calibration for regression uncertainty calibration. Let $\mathcal{P}$ denote the set of all possible probability distributions, so that $\Pi_R \in \mathcal{P}$. A probabilistic forecaster is distribution-calibrated if the predicted probability distribution $f_{\hat{R}}$ matches the observed (error)

[1] For notational simplicity, we further refer to $\chi^2_L(\tau)$ as the percent point function (inverse CDF) of a χ^2-distributed random variable with L degrees of freedom and quantile $\tau \in [0, 1]$.

distribution Π_R given a certain probability distribution π [41], so that

$$f_{\hat{R}}(\hat{r}|\Pi_R = \pi) = \pi(\hat{r}), \tag{5.7}$$

must hold for all $\pi \in \mathcal{P}$ and for all $\hat{r} \in \mathcal{R}$ [41]. For example, consider a probabilistic forecaster that predicts several samples with equal distributions (e.g., with same mean and variance using Gaussian distributions). With these predictions, it is possible to construct an error distribution for the corresponding ground-truth samples $\bar{R}$ as well. If the predicted distributions match the observed ones, a probabilistic forecaster is distribution-calibrated [41]. This principle is schematically shown in Fig. 5.3ii. The authors in [41] also point out that a distribution-calibrated forecaster is also quantile-calibrated [41]. In the context of object detection, false positive predictions may occur which, however, do not have a corresponding ground-truth label. Thus, we can not compute error statistics for these predictions so that they are discarded in this process.

The authors in [41] use Negative Log Likelihood (NLL) for measuring distribution calibration. The authors decompose the NLL into a calibration and a refinement loss and show the benefits of distribution calibration to the former calibration loss. Although the definition for distribution calibration is more restrictive and constructed in the sense of the well-known definition for semantic confidence calibration, the set of possible probability distributions $\mathcal{P}$ can be very large and intractable if it is not restricted to a parametric distribution family. This leads us to the next definition for regression uncertainty calibration.

5.2.3 Variance Calibration

Recently, the authors in [42] and [18] independently introduced the definition for variance calibration which is designed to evaluate the calibration properties of Gaussian distributions. Given a joint ground-truth data distribution $f_{\mathbf{X},\bar{R}}(\mathbf{x}, \bar{r})$ with input images $\mathbf{X} \in \mathcal{X}$ and ground-truth bounding box positions $\bar{R} \in \mathcal{R}$, a probabilistic forecaster is variance-calibrated if the predicted variance matches the observed one given a certain variance level [42, 18], so that

$$\mathbb{E}_{\mathbf{X},\bar{R}}\left[(\bar{r} - \mu_{\hat{R}|\mathbf{X}})^2 | \sigma^2_{\hat{R}|\mathbf{X}} = \sigma^2\right] = \sigma^2, \tag{5.8}$$

must hold for all $\sigma^2 \in \mathbb{R}_{>0}$ [42, 18]. We further refer to $\mathcal{N}(\mu_{\bar{R}}, \sigma^2_{\bar{R}})$ as the observed (error) distribution with mean $\mu_{\bar{R}} \in \mathbb{R}$ and observed variance $\sigma^2_{\bar{R}} \in \mathbb{R}_{>0}$. A variance-calibrated forecaster is also quantile-calibrated if the observed error distribution is a Gaussian and the forecaster is not biased in its predictions. The error distribution is then equal to the predicted normal distribution $\mathcal{N}(\mu_{\hat{R}|\mathbf{X}}, \sigma^2_{\hat{R}|\mathbf{X}})$ which results in equal quantile functions $F^{-1}_{\hat{R}|\mathbf{X}}(\tau) = F^{-1}_{\bar{R}}(\tau)$ so that the predicted quantiles match the observed ones. In application, the observed variance is equal to the Mean Squared Error (MSE) for a certain predicted variance level. Furthermore, if the observed variance does not depend on the predicted mean, i.e., $\mathrm{Cov}(\mu_{\hat{R}|\mathbf{X}}, \sigma^2_{\bar{R}}) = 0$, then a variance-calibrated forecaster is also distribution-calibrated as the set of probability distributions $\mathcal{P}$ is restricted to normal distributions and the mean $\mu_{\bar{R}}$ is no influential factor. Therefore, variance calibration can be interpreted as a variant of distribution calibration for normal distributions with additional requirements

on the forecaster and the ground-truth data. This principle is schematically shown in Fig. 5.3iii.

To test for variance calibration, the authors in [42] propose the Expected Normalized Calibration Error (ENCE) which measures the unweighted and normalized difference between predicted and observed standard deviation for a certain standard deviation level. Similar to the Expected Calibration Error (ECE) from semantic confidence calibration, the ENCE utilizes a binning scheme over the predicted standard deviation with I distinct bins to measure calibration by means of the predicted standard deviation, so that the ENCE is defined by

$$\text{ENCE} := \frac{1}{I}\sum_{i=1}^{I}\frac{|\text{RMSE}(i) - \text{RMV}(i)|}{\text{RMV}(i)}, \tag{5.9}$$

where $\text{RMSE}(i)$ and $\text{RMV}(i)$ denote the root mean squared error and the root mean variance within bin i, respectively. In practice, we choose the maximum of the estimated standard deviation σ_{max} and divide the interval $[0, \sigma_{\text{max}}]$ into I equally sized bins. Similarly, the authors in [18] propose the Uncertainty Calibration Error (UCE) which measures the weighted and unnormalized difference between predicted and observed variance for a certain variance level. Thus, the UCE is defined by

$$\text{UCE} := \sum_{i=1}^{I}\frac{N_i}{N}|\text{MSE}(i) - \text{MV}(i)|, \tag{5.10}$$

where $\text{MSE}(i)$ and $\text{MV}(i)$ denote the mean squared error and the mean variance within bin i, respectively [18]. The advantage of the ENCE is that the miscalibration can be quantified relative to the predicted uncertainty. Therefore, the ENCE is independent of the size of the investigation space $\mathcal{R}$.

A drawback of both metrics is that they can not capture the properties of multivariate normal distributions with possible correlations. Using these formulations, it is not straightforward to measure calibration by means of predicted covariances. Furthermore, our recently derived M-QCE metric only measures the marginal calibration error which is not sensitive to calibration by means of different variances. Therefore, we first consider the Standardized Generalized Variance (SGV) as a property of a multivariate normal distribution which is defined by $\sigma^2_{\text{SG}} = \det(\boldsymbol{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}})^{\frac{1}{L}}$ [111, 112]. The SGV reflects the dispersion of a distribution over all dimensions L. This allows distributions with similar dispersion to be grouped and compared to each other. The distribution $f_{\sigma^2_{\text{SG}}}$ of the SGV is directly connected to the output of the object detector as it is derived by the random variable $\hat{\mathbf{R}}$.

In (5.5), we already proposed the M-QCE to evaluate the HPDR of multivariate Gaussian distributions. However, the M-QCE evaluates the calibration properties of a forecaster over all predictions. Instead, we seek to evaluate a forecaster conditioned on its predictions, similar to the ECE known from semantic confidence calibration evaluation (cf. Sec. 3.2.1). The ECE is conditioned on the predicted confidence to measure miscalibration by means of the model output. Similarly, we can measure the quantile coverage conditioned

on the SGV as the model output by

$$\mathbb{E}_{\sigma^2_{\text{SG}} \sim f_{\sigma^2_{\text{SG}}}} \Big[\big| \mathbb{P}\big(\epsilon_{\hat{\mathbf{R}}|\mathbf{X}} \leq \chi^2_L(\tau) | \sigma^2_{\text{SG}}\big) - \tau \big| \Big]. \tag{5.11}$$

We can now reformulate the M-QCE to the Conditional Quantile Calibration Error (C-QCE) which measures the error between predicted quantile and observed quantile coverage as a function of the model output given by the SGV. In practice, we further use the square root of the SGV to achieve a better data distribution of the dispersion for binning. Similar to the UCE and ENCE, a binning scheme with I bins over the square root of the SGV is applied, so that the C-QCE is an approximation of (5.11) on a finite data set $\mathcal{D}$ given by

$$\text{C-QCE}(\tau) := \sum_{i=1}^{I} \frac{N_i}{N} |\text{freq}(i) - \tau|, \tag{5.12}$$

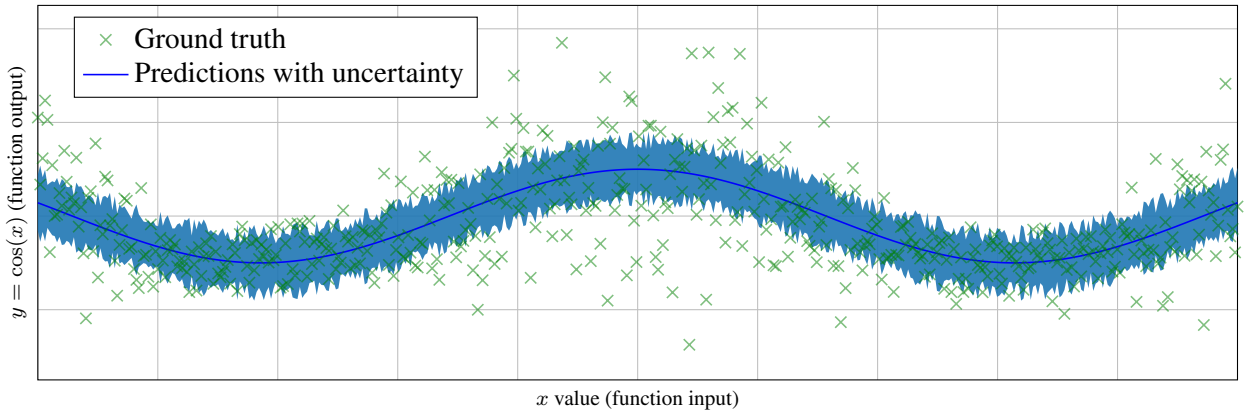
where

$$\text{freq}(i) = \frac{1}{N_i} \sum_{n \in \mathcal{M}_i} \mathbb{1}\big(\epsilon_{\hat{\mathbf{R}}|\mathbf{x}_n} \leq \chi^2_L(\tau)\big), \tag{5.13}$$

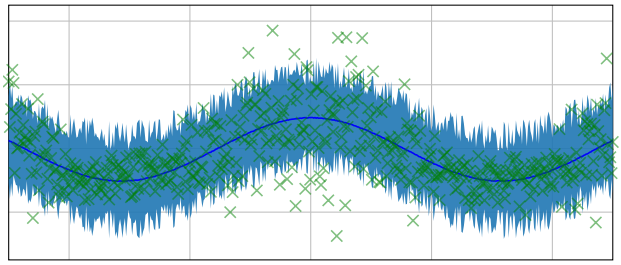
denotes the prediction interval coverage frequency within bin i, where $\mathcal{M}_i$ is the set of sample indices with all samples falling into bin i. In the following, these metrics are used for uncertainty calibration evaluation.

5.3 Review of Methods for Regression Uncertainty Calibration

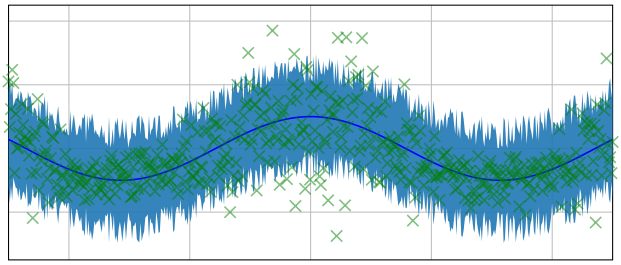
In this section, we present state-of-the-art calibration methods such as Isotonic Regression [34], Variance Scaling [42, 18], and GP-Beta [41]. These techniques perform calibration w.r.t. one of the previously introduced definitions for regression uncertainty calibration. The effects of calibration by means of individual calibration targets are qualitatively shown in Fig. 5.4. We can divide these calibration techniques into parametric and non-parametric methods. The former methods yield a parametric probability distribution (e.g., Gaussian) as calibration output, whereas the latter ones yield probability distributions of arbitrary shape. While the non-parametric distributions are more flexible in representing any data distributions, it might be necessary to utilize a parametric distributions after calibration, e.g., for subsequent applications such as Kalman filtering for object tracking (cf. Chap. 6). Therefore, we propose an extension to the existing calibration methods that is able to perform parametric uncertainty calibration in the sense of distribution calibration. We further derive a calibration scheme for a joint multivariate uncertainty recalibration of multiple dimensions. Finally, we adapt the Gaussian process scheme for a covariance estimation and recalibration. This allows for a post-hoc introduction of correlations between independently inferred probability distributions. Note that we assert normal distributions as the input to the calibration methods as we work with the output of probabilistic detection models that have been introduced in Sec. 2.1. We start by reviewing the state-of-the-art methods for regression uncertainty calibration and present our extensions for parametric and joint uncertainty calibration that are subject of our publication in [6].



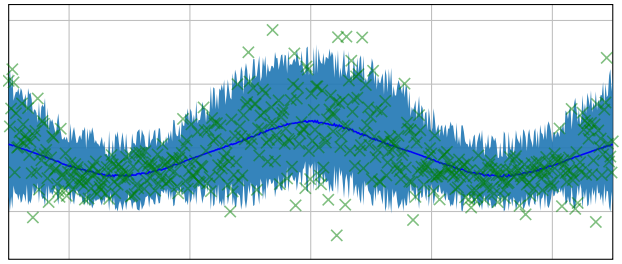
(i) Artificial data (green) and uncalibrated estimator with according predicted uncertainty (blue).



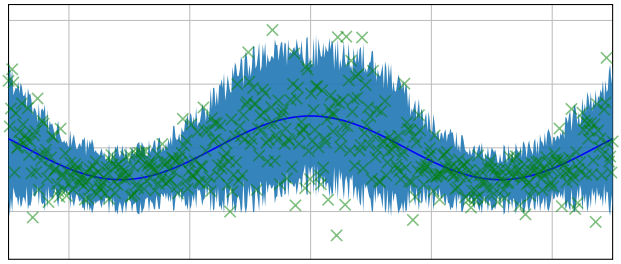
(ii) Isotonic Regression [34].



(iii) Variance Scaling [42, 18].



(iv) GP-Beta [41].



(v) GP-Normal [6].

Figure 5.4: Qualitative example on artificial data to demonstrate the differences between methods for quantile calibration (ii), variance calibration (iii), and distribution calibration (iv & v) [6, p. 7, Fig. 2]. The ground-truth data is obtained by sampling from a cosine with aleatoric Gaussian noise (green points). The noise amplitude is proportional to the y-value of the cosine function (with a small offset), thus, the aleatoric uncertainty is correlated with the function value. Furthermore, we assume an unbiased estimator (blue) with randomly sampled variance for each point which, however, is equally sampled for the whole function. In this example, we can see that the methods for quantile calibration (ii) as well as for variance calibration (iii) can not to capture the dependency between aleatoric uncertainty and the function value. In contrast, the methods for distribution calibration (iv & v) are able to recalibrate the predicted uncertainty scores.

5.3.1 Non-Parametric Calibration

A non-parametric calibration method takes an uncalibrated probability distribution and outputs a calibrated distribution that has no analytical form, i.e., the PDF is not fixed to any arbitrary shape. The advantage of this distribution representation is that calibration methods are not restricted to any assumptions and thus can flexibly represent any data distribution. The existing non-parametric calibration methods are designed to output a non-parametric density function given any probability distribution as input to fit the calibrated distribution to the observed (error) distribution. Commonly, the calibration techniques are applied on the univariate CDF of the input distributions (for each dimension independently) which denotes the cumulative probability mass of a certain point. Since the total probability mass of any distribution is 1, the CDF also always outputs scores only within the $[0, 1]$ interval. This allows for the application of calibration techniques known from semantic confidence calibration, since these methods are bound to the $[0, 1]$ interval as well. Specifically, the Isotonic Regression [85, 34] as well as the Beta Calibration [43, 41] methods have recently been adapted for regression calibration which are presented in the following.

Isotonic Regression

As already described in Sec. 5.2.1, the same authors in [34] propose a recalibration framework to achieve a quantile-calibrated forecaster. We further denote this as marginal calibration as the target is to construct a recalibration method that leads to a quantile-calibrated model where only the marginal calibration properties over all given samples are of interest. Let $f_{\hat{R}}(\hat{r})$ and $F_{\hat{R}|\mathbf{X}}(\hat{r})$ denote the PDF and CDF of the predicted object locations in a single dimension, respectively. For any quantile level $\tau \in [0, 1]$, the authors in [34] seek to estimate the true probability of $\mathbb{P}\big(\bar{R} \leq F^{-1}_{\hat{R}|\mathbf{X}}(\tau)\big)$ $\big($cf. (5.1)$\big)$ given a finite data set. If we observe a deviation between the estimated prediction intervals and observed interval coverage (using the true target score), a calibration function h will be necessary to recalibrate the estimated quantile boundaries. As the recalibration target, the authors utilize the Empirical Cumulative Density Function (ECDF) which can be interpreted as an estimator for the probability distribution of the true underlying data generation process. The ECDF is a step function that denotes, for each sample n, the fraction of observations whose values are less than or equal to the actual sample. According to the Glivenko–Cantelli theorem [113, 114], the ECDF converges to the underlying distribution as the number of observations grows. In the case of quantile calibration, we are interested in the probability that an observation falls into the estimated prediction interval. The authors in [34] utilize the ECDF to represent the according probability distribution. Given N observations with known (univariate) ground-truth bounding box position $\bar{r}_n$ as well as the estimated CDF $F_{\hat{R}|\mathbf{x}_n}$ for each sample (obtained by a probabilistic object detector), the ECDF can be constructed by

$$F_{\text{emp}}(\bar{r}_n) = \frac{1}{N} \sum_{n^*=1}^{N} \mathbb{1}\Big(F_{\hat{R}|\mathbf{x}_{n^*}}(\bar{r}_{n^*}) \leq F_{\hat{R}|\mathbf{x}_n}(\bar{r}_n)\Big). \tag{5.14}$$

By constructing the ECDF for the probability distribution of the prediction interval coverage, the inverse empirical quantile function reflects the desired estimate for $\mathbb{P}\big(\bar{R} \leq F^{-1}_{\hat{R}|\mathbf{X}}(\tau)\big)$ which is the final target for

quantile calibration. Thus, matching the predicted CDF scores with the according ECDF counterpart finally yields in quantile-calibrated estimates.

For calibration, the function $h(\cdot)$ serves as a mapping from the uncalibrated CDF to a calibrated one, so that $h : [0,1] \rightarrow [0,1]$. The authors in [34] propose to use the Isotonic Regression calibration method known from semantic confidence calibration [85]. As originally proposed by the authors in [85], Isotonic Regression fits a piece-wise constant and monotonically increasing function as recalibration function to map uncalibrated confidence estimates to calibrated ones. Thus, Isotonic Regression can be seen as a variant of Histogram Binning [36], but with flexible bin sizes and a flexible amount of bins. A training data set for the Isotonic Regression method can be constructed using the predicted CDF $F_{\hat{R}|\mathbf{x}_n}(\bar{r}_n)$ as the input to the regression function and the according ECDF score $F_{\text{emp}}(\bar{r}_n)$ as the regression target for each sample n with ground-truth $\bar{r}_n$, so that $\mathcal{D} = \{F_{\hat{R}|\mathbf{x}_n}(\bar{r}_n), F_{\text{emp}}(\bar{r}_n)\}_{n=1}^{N}$. We can use the training set $\mathcal{D}$ to construct the regression function which serves as a mapping from uncalibrated quantiles to calibrated ones [34].

During inference, the mapping function is used to transform the CDF of each input data individually. For each input sample, we draw a number of T points denoted by $\mathcal{D}_n^* = \big\{\big(r_t, F_{\hat{R}|\mathbf{x}_n}(r_t)\big)\big\}_{t=1}^{T}$ to represent the predicted CDF of the input data with index $n \in \{1, \dots, N\}$. Afterwards, we can pass each of these points through the calibration mapping to finally yield a non-parametric representation of the calibrated cumulative. Thus, the calibrated CDF $G_{\hat{R}|\mathbf{X}}(\hat{r})$ for each point in $\mathcal{D}_n^*$ is given by

$$G_{\hat{R}|\mathbf{x}_n}(r_{t,n}) = h\big(F_{\hat{R}|\mathbf{x}_n}(r_{t,n})\big), \quad \forall t \in \{1, \dots, T\}. \tag{5.15}$$

An approximation for the respective calibrated density scores $g_{\hat{R}|\mathbf{x}_n}(r_{t,n})$ at the point locations $r_{t,n}$ can be obtained by differentiation for each sample with index n.

Beta Calibration with Gaussian Process Parameter Estimation

Similarly to Isotonic Regression, the authors in [41] adapted the Beta Calibration method [43] from the scope of confidence calibration and applied it to regression uncertainty calibration (cf. equation (3.41 in Sec. 3.3.2 for a detailed description of Beta Calibration). Let h_β denote the Beta Calibration function which transforms the quantiles $\tau \in [0,1]$ (obtained by the CDF) to calibrated ones. For regression calibration, the Beta Calibration parameters $a, b \in \mathbb{R}_{>0}$ and $c \in \mathbb{R}$ are used to rescale the cumulative of the input distribution [41], so that the calibrated CDF is given by

$$G_{\hat{R}|\mathbf{X}}(\hat{r}) = \Phi\Big(a \cdot \log\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big) - b \cdot \log\big(1 - F_{\hat{R}|\mathbf{X}}(\hat{r})\big) + c\Big) \tag{5.16}$$

$$= \Phi\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big) \tag{5.17}$$

$$= h_\beta\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big), \tag{5.18}$$

with $\Phi(\cdot)$ as the sigmoid function, where

$$z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big) = a \cdot \log\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big) - b \cdot \log\big(1 - F_{\hat{R}|\mathbf{X}}(\hat{r})\big) + c. \tag{5.19}$$

Similar to the previous Isotonic Regression [34], the Beta Calibration function seeks to serve as a transformation of uncalibrated quantiles to calibrated ones. Since the PDF can also be interpreted as the derivative of the CDF, it is possible to derive the PDF by differentiation [41], so that the calibrated PDF $g_{\hat{R}|\mathbf{X}}$ is given by

$$g_{\hat{R}|\mathbf{X}}(\hat{r}) = \frac{\mathrm{d}h_\beta\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big)}{\mathrm{d}\hat{r}} \tag{5.20}$$

$$= \frac{\mathrm{d}h_\beta\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big)}{\mathrm{d}z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)} \frac{\mathrm{d}z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)}{\mathrm{d}F_{\hat{R}|\mathbf{X}}(\hat{r})} \frac{\mathrm{d}F_{\hat{R}|\mathbf{X}}(\hat{r})}{\mathrm{d}\hat{r}} \tag{5.21}$$

$$= h'_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big) \cdot f_{\hat{R}|\mathbf{X}}(\hat{r}), \tag{5.22}$$

where $\frac{\mathrm{d}F_{\hat{R}|\mathbf{X}}(\hat{r})}{\mathrm{d}\hat{r}}$ reduces to the uncalibrated input PDF and

$$h'_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big) = \frac{\mathrm{d}h_\beta\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big)}{\mathrm{d}z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)} \frac{\mathrm{d}z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)}{\mathrm{d}F_{\hat{R}|\mathbf{X}}(\hat{r})}, \tag{5.23}$$

is the derivative of h_β w.r.t. $\hat{r}$ [41]. The derivative of the sigmoid function can be expressed by $\Phi(z_\beta)' = \Phi(z_\beta)\big(1 - \Phi(z_\beta)\big)$, so that the function h'_β can be derived by

$$h'_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big) = \frac{\mathrm{d}h_\beta\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big)}{\mathrm{d}F_{\hat{R}|\mathbf{X}}(\hat{r})} = \frac{\mathrm{d}h_\beta\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big)}{\mathrm{d}z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)} \frac{\mathrm{d}z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)}{\mathrm{d}F_{\hat{R}|\mathbf{X}}(\hat{r})} \tag{5.24}$$

$$= \Phi\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big)\Big[1 - \Phi\Big(z_\beta\big(F_{\hat{R}|\mathbf{X}}(\hat{r})\big)\Big)\Big]\left[\frac{a}{F_{\hat{R}|\mathbf{X}}(\hat{r})} + \frac{b}{1 - F_{\hat{R}|\mathbf{X}}(\hat{r})}\right] \tag{5.25}$$

$$= G_{\hat{R}|\mathbf{X}}(\hat{r})\Big(1 - G_{\hat{R}|\mathbf{X}}(\hat{r})\Big)\left[\frac{a}{F_{\hat{R}|\mathbf{X}}(\hat{r})} + \frac{b}{1 - F_{\hat{R}|\mathbf{X}}(\hat{r})}\right]. \tag{5.26}$$

If the calibration parameters were trained by standard Maximum Likelihood Estimation (MLE) using the NLL as the loss function, this calibration method would also perform marginal calibration yielding quantile-calibrated distributions. This corresponds to the training procedure used within marginal recalibration which we already introduced in the last section. However, the authors in [41] propose the term of distribution calibration and thus seek to derive a calibration function that applies uncertainty calibration with a specific parameter set for each input distribution, individually. As opposed to the training of the Isotonic Regression method [34], we do not compute an empirical CDF over the training data set $\mathcal{D}$ as this would result in

marginal recalibration. Instead, the authors in [41] use a Gaussian Process (GP) for inferring the calibration parameters $a(\cdot), b(\cdot)$ and $c(\cdot)$ as a function of the provided sample. The authors propose to use three latent functions $w_a(\cdot)$, $w_b(\cdot)$, and $w_c(\cdot)$ which are directly mapped to the calibration parameters $a(\cdot), b(\cdot)$ and $c(\cdot)$ by

$$\begin{aligned} a(\cdot) &= \exp(\nu_a w_a(\cdot) + \kappa_a), \\ b(\cdot) &= \exp(\nu_b w_b(\cdot) + \kappa_b), \\ c(\cdot) &= \nu_c w_c(\cdot) + \kappa_c, \end{aligned} \tag{5.27}$$

where the exponential functions guarantee that $a, b \in \mathbb{R}_{>0}$. The advantage of using the intermediate functions w_a, w_b, and w_c is that we have no restrictions on the GP, so that the latent functions can directly be drawn from the GP. Furthermore, each additional scaling factor $\nu_a, \nu_b, \nu_c \in \mathbb{R}$ and bias $\kappa_a, \kappa_b, \kappa_c \in \mathbb{R}$ are used to prevent possible distortion during the GP initialization phase [41].

The idea of using a GP for distribution calibration is that neighboring samples in the input space (with similar mean and variance) are likely to produce similar outputs. The concept of distribution calibration using a GP model is to construct an error distribution for each sample individually based on the local neighborhood. These error distributions can be compared to the ones that have been estimated by the object detector, so that the GP can finally be used for a recalibration to achieve distribution calibration. For this reason, the authors in [41] introduce a multi-output GP [115, 116, 117] to infer the latent functions $w_a(\cdot)$, $w_b(\cdot)$, and $w_c(\cdot)$ by means of the uncalibrated input. We further refer to the function weight vector as $\mathbf{w} = (w_a, w_b, w_c)^\top$. A GP is a stochastic process such that a finite collection of random variables follows a joint multivariate normal distribution. The GP is parameterized by a mean and a positive semidefinit kernel or covariance function. Especially the kernel function $k(\cdot, \cdot)$ is important as it is used to construct the covariance matrix of the GP which reflects the correlations between the random variables. Common kernel functions for vector-valued inputs are the squared exponential kernel, the rational quadratic kernel, or the periodic kernel.

However, in our case, the input to the kernel function are the (uncalibrated) predictions of a probabilistic object detector that consist of the estimated distribution parameters for a Gaussian distribution. Thus, the authors in [41] utilize a univariate Gaussian embedding using the Radial Basis Function (RBF) kernel [118] which is given by

$$k\big((\mu_i, \sigma_i^2), (\mu_j, \sigma_j^2)\big) = \frac{\theta}{|\sigma_{ij}^2|^{\frac{1}{2}}} \exp\left(-\frac{1}{2\sigma_{ij}^2}(\mu_i - \mu_j)^2 \right), \tag{5.28}$$

with length scale parameter $\theta \in \mathbb{R}_{>0}$, where $\sigma_{ij}^2 = \sigma_i^2 + \sigma_j^2 + \theta^2$ [41]. Unfortunately, the authors in [118] do not provide a mathematical proof that this kernel function yields a positive semidefinit covariance matrix. However, we conducted extensive studies so that we have been able to empirically verify that the resulting covariance matrices are valid.

Let $\mathcal{D} = \left\{ \left(\bar{r}_n, \mu_{\hat{R}|\mathbf{x}_n}, \sigma^2_{\hat{R}|\mathbf{x}_n} \right) \right\}_{n=1}^{N}$ denote the training set with N samples consisting of the ground-truth

$\bar{r}_n$ as well as of the predicted mean $\mu_{\hat{R}|\mathbf{x}_n}$ and variance $\sigma^2_{\hat{R}|\mathbf{x}_n}$. The predicted mean and variance are both obtained by a probabilistic object detector. Note that we are working with the univariate (independent) bounding box quantities. Furthermore, we denote the mean vector by $\boldsymbol{\mu}_\mathcal{D} = (\mu_{\hat{R}|\mathbf{x}_1}, \dots, \mu_{\hat{R}|\mathbf{x}_N})^\top$ and the variance vector by $\boldsymbol{\sigma}^2_\mathcal{D} = (\sigma^2_{\hat{R}|\mathbf{x}_1}, \dots, \sigma^2_{\hat{R}|\mathbf{x}_N})^\top$ which both hold the predictions of the whole data set $\mathcal{D}$. Using the kernel function in (5.28), we denote the covariance matrix $\mathbf{K}_\mathcal{D} \in \mathbb{R}^{N\times N}$ which is given by $\mathbf{K}_\mathcal{D} = k\big((\boldsymbol{\mu}_\mathcal{D}, \boldsymbol{\sigma}^2_\mathcal{D}), (\boldsymbol{\mu}_\mathcal{D}, \boldsymbol{\sigma}^2_\mathcal{D})'\big)$. Moreover, we denote the respective function weight matrix by $\mathbf{W}_\mathcal{D} = \big(\mathbf{w}_1^\top, \dots, \mathbf{w}_N^\top\big)^\top$ that contains the latent functions for all samples in $\mathcal{D}$, where $\mathbf{W}_\mathcal{D} \in \mathcal{W}$. The matrix $\mathbf{W}_\mathcal{D}$ is also a function of the input samples, which we will omit in the following for notation simplicity.

In our case, when modeling $\mathbf{W}_\mathcal{D}$ using a GP with zero mean and covariance matrix $\mathbf{K}_\mathcal{D}$, the latent functions follow a joint multivariate normal distribution, so that

$$f(\mathbf{W}_\mathcal{D}|\mathcal{D}) = \mathcal{N}(\mathbf{W}_\mathcal{D}|0, \mathbf{K}_\mathcal{D} \otimes \mathbf{B}_\beta), \tag{5.29}$$

where $\otimes$ is the Kronecker product and $\mathbf{B}_\beta \in \mathbb{R}^{3\times 3}$ is the coregionalization matrix that captures the dependencies between all latent functions. The target is to obtain a vector-valued output for each sample (the weights $w_a(\cdot)$, $w_b(\cdot)$, and $w_c(\cdot)$). Such a multi-output GP is also called an intrinsic coregionalization model (ICM) [119] which models the dependencies between the outputs as a linear combination of multiple latent variables that share the same covariance kernel. Thus, for a single sample, the weights are drawn by the GP model so that

$$w_a, w_b, w_c \sim \text{gp}(0, k(\cdot,\cdot), \mathbf{B}_\beta), \tag{5.30}$$

where $\text{gp}(0, k(\cdot,\cdot), \mathbf{B}_\beta)$ denotes the GP model with zero mean, kernel function k and coregionalization matrix $\mathbf{B}_\beta$. In summary, the GP model uses the scaling weights ν_a, ν_b, ν_c, the scaling biases $\kappa_a, \kappa_b, \kappa_c$, the coregionalization matrix $\mathbf{B}_\beta$ and the length scale parameter θ as trainable parameters.

In practice, the authors in [41] use an approximate variational GP for regression uncertainty calibration. We can not use the standard log marginal likelihood for the training of the GP parameters as we use intermediate latent functions that are passed through a non-linear exponential function and that do not have a ground-truth [41]. Instead, it is necessary to utilize the posterior predictive distribution from Bayesian statistics to obtain an appropriate likelihood function. Given the observations $\bar{\mathbf{r}}_\mathcal{D} = (\bar{r}_1, \dots, \bar{r}_N)^\top$ for the ground-truth bounding box positions in the data set $\mathcal{D}$ (for a single dimension), the model likelihood is given by

$$f(\bar{\mathbf{r}}_\mathcal{D}|\mathcal{D}) = \int_\mathcal{W} f(\bar{\mathbf{r}}_\mathcal{D}|\mathbf{W}_\mathcal{D}, \mathcal{D}) f(\mathbf{W}_\mathcal{D}|\mathcal{D}) \mathrm{d}\mathbf{W}_\mathcal{D} \tag{5.31}$$

$$= \int_\mathcal{W} \prod_{n=1}^{N} \Big[f\big(\bar{r}_n|\mathbf{w}_n, \mu_{\hat{R}|\mathbf{x}_n}, \sigma^2_{\hat{R}|\mathbf{x}_n}\big) \Big] f\big(\mathbf{W}_\mathcal{D}|\mu_{\hat{R}|\mathbf{x}_n}, \sigma^2_{\hat{R}|\mathbf{x}_n}\big) \mathrm{d}\mathbf{W}_\mathcal{D}, \tag{5.32}$$

where

$$f\big(\bar{r}_n|\mathbf{w}_n, \mu_{\hat{R}|\mathbf{x}_n}, \sigma^2_{\hat{R}|\mathbf{x}_n}\big) = f_{\hat{R}|\mathbf{x}_n}(\bar{r}_n) \cdot h'_\beta\big(F_{\hat{R}|\mathbf{x}_n}(\bar{r}_n)\big), \tag{5.33}$$

is the likelihood obtained using the link function $h'_\beta(\cdot)$ with weights $\mathbf{w}_n$ [41]. Furthermore, the term $f\big(\mathbf{W}_\mathcal{D}|\mu_{\hat{R}|\mathbf{x}_n}, \sigma^2_{\hat{R}|\mathbf{x}_n}\big)$ is the Gaussian likelihood of the GP model itself.

The GP is approximate because the kernel matrix, which is constructed during the computation of the GP, scales quadratically with the amount of training samples which yields the complexity $\mathcal{O}(N^2)$. Therefore, the authors in [41] use a scalable inference scheme [120, 121] where a set of N^* inducing points is learned to represent the data set so that each inducing point has an own mean μ_u and variance σ_u^2 that needs to be learned. We refer to [120], [121], and [41] for a detailed discussion about approximate GP models.

With this method, it is finally possible to perform distribution calibration by mapping uncalibrated Gaussians to calibrated non-parametric probability distributions. Similar to the previously described Isotonic Regression method [34], we start by generating a set of T points that describe the uncalibrated PDF and CDF denoted by $\mathcal{D}_n^* = \big\{\big(r_t, f_{\hat{R}|\mathbf{x}_n}(r_t), F_{\hat{R}|\mathbf{x}_n}(r_t)\big)\big\}_{t=1}^T$. Furthermore, we draw a set of weights from the trained GP model given the new inference data. The calibrated PDF and CDF are approximated for each point in $\mathcal{D}_n^*$ by computing the average of the rescaled PDF and CDF estimates using h'_β and h_β in (5.20) and (5.16), respectively, given the sampled set of weights. In this way, we can finally construct the recalibrated and non-parametric probability distribution for each sample during inference.

5.3.2 Parametric Calibration

In contrast to the previously introduced non-parametric calibration techniques, parametric methods seek for a recalibration using a known analytical representation of the probability distribution. In this context, we present the Variance Scaling which is a parametric calibration method for normal distributions.

Variance Scaling

The Variance Scaling method for variance calibration has independently been introduced by [42] and [18] and can be interpreted as a kind of temperature scaling [13] from classification calibration for the uncalibrated variances of normal distributions. In this way, a single scaling parameter $w_\sigma \in \mathbb{R}_{>0}$ is learned to yield a calibrated normal distribution by

$$g_{\hat{R}|\mathbf{X}}(\hat{r}) = \mathcal{N}\Big(\hat{r}; \mu_{\hat{R}|\mathbf{X}}, \big(w_\sigma \cdot \sigma_{\hat{R}|\mathbf{X}}\big)^2\Big), \tag{5.34}$$

where the scaling parameter is a fixed constant after calibration training [42, 18]. The parameter is trained using MLE with the NLL as the optimization objective by

$$\mathcal{L}(w_\sigma) = -\sum_{n=1}^{N} \log\left[\frac{1}{\sqrt{2\pi}(w_\sigma \cdot \sigma_{\hat{R}|\mathbf{x}_n})} \exp\Big(-\frac{1}{2}(w_\sigma \cdot \sigma_{\hat{R}|\mathbf{x}_n})^{-2}(\bar{r}_n - \mu_{\hat{R}|\mathbf{x}_n})^2\Big)\right] \tag{5.35}$$

$$\propto -N\log(w_\sigma) - \frac{1}{2w_\sigma^2}\sum_{n=1}^{N}\sigma_{\hat{R}|\mathbf{x}_n}^{-2}(\bar{r}_n - \mu_{\hat{R}|\mathbf{x}_n})^2, \tag{5.36}$$

which is to be minimized. Thus, we seek to get the minimum of the optimization objective which can analytically be determined using its derivative $\frac{\mathrm{d}\mathcal{L}(w_\sigma)}{\mathrm{d}w_\sigma} = 0$ [18], so that

$$-\frac{N}{w_\sigma} + \frac{1}{w_\sigma^3}\sum_{n=1}^{N}\sigma_{\hat{R}|\mathbf{x}_n}^{-2}(\bar{r}_n - \mu_{\hat{R}|\mathbf{x}_n})^2 = 0 \tag{5.37}$$

$$\Leftrightarrow \quad -Nw_\sigma^2\sum_{n=1}^{N}\sigma_{\hat{R}|\mathbf{x}_n}^{-2}(\bar{r}_n - \mu_{\hat{R}|\mathbf{x}_n})^2 = 0 \tag{5.38}$$

$$\Leftrightarrow \quad w_\sigma = \pm\sqrt{\frac{1}{N}\sum_{n=1}^{N}\sigma_{\hat{R}|\mathbf{x}_n}^{-2}(\bar{r}_n - \mu_{\hat{R}|\mathbf{x}_n})^2} \tag{5.39}$$

gives us the analytical solution for w_σ given a certain set $\mathcal{D}$ of training samples [18], where the data set $\mathcal{D} = \big\{\big(\bar{r}_n, \mu_{\hat{R}|\mathbf{x}_n}, \sigma^2_{\hat{R}|\mathbf{x}_n}\big)\big\}_{n=1}^{N}$ with N samples consists of the ground-truth $\bar{r}_n$ as well as of the predicted mean $\mu_{\hat{R}|\mathbf{x}_n}$ and variance $\sigma^2_{\hat{R}|\mathbf{x}_n}$.

5.4 Parametric Calibration using Gaussian Processes

On the one hand, we seek to keep parameterized probability distributions after calibration. Although non-parametric distributions might have a better representational power of the underlying data distribution, it is often advantageous to use parametric distributions especially for subsequent processes such as object tracking (cf. Chap. 6). On the other hand, we seek for more flexibility during calibration compared to the previously introduced Variance Scaling [42, 18] to finally achieve distribution calibration. For this reason, we adapt the recalibration framework provided by [41] which is based on the GP recalibration. Instead of a non-parametric Beta Calibration on the CDF, we also adapt the Variance Scaling scheme by [42, 18] but seek to obtain the scaling parameter $w_\sigma(f_{\hat{R}|\mathbf{X}})$ by a GP as a function of the input distribution $f_{\hat{R}|\mathbf{X}}$ to the calibration method, so that

$$\log\big(w_\sigma(\cdot)\big) \sim \mathrm{gp}\big(0, k(\cdot,\cdot)\big), \tag{5.40}$$

which finally yields a similar recalibrated PDF as within (5.34) but with an input-dependent scaling weight

$$g_{\hat{R}|\mathbf{X}}(\hat{r}) = \mathcal{N}\Big(\hat{r}; \mu_{\hat{R}|\mathbf{X}}, \big(w_\sigma(f_{\hat{R}|\mathbf{X}}) \cdot \sigma_{\hat{R}|\mathbf{X}}\big)^2\Big). \tag{5.41}$$

This method is further denoted by GP-Normal. We use the exponential function to guarantee $w_\sigma(f_{\hat{R}|\mathbf{X}}) \in \mathbb{R}_{>0}$. Since we only infer a single parameter, a coregionalization matrix $\mathbf{B}$ is not necessary any more. This method offers the flexibility of a GP towards distribution calibration and preserves a parametric Gaussian distribution for subsequent processes. Therefore, the GP-Normal can be seen as an extension of the definition for variance calibration towards distribution calibration but for normally distributed data. Thus, under the assumption of normally distributed data, the GP-Normal seeks to achieve

$$\mathbb{E}_{\mathbf{X},\bar{R}}\big[(\bar{R} - \mu_{\hat{R}|\mathbf{X}})^2 \big| \Pi_{\bar{R}} = \mathcal{N}(\mu, \sigma^2)\big] = \sigma^2, \tag{5.42}$$

for all $\mu \in \mathbb{R}$ and for all $\sigma^2 \in \mathbb{R}_{>0}$. The concept for the GP-Normal method is schematically shown in Fig. 5.5. Basically, we can plug in any desired parametric probability distribution. For our probabilistic object detection model from the experiments in Sec. 5.5, we observe error distributions between the predicted mean and the ground-truth scores that are rather Cauchy-distributed than normally distributed. This is shown in Fig. 5.6. Therefore, we can also use the GP to construct a calibrated Cauchy distribution with location $\vartheta \in \mathbb{R}$ and scale $\gamma \in \mathbb{R}_{>0}$, where the location parameter is approximated using the uncalibrated mean $\mu_{\hat{R}|\mathbf{X}}$ and the scale parameter is obtained by

$$\log\big(w_\gamma(\cdot)\big) \sim \mathrm{gp}\big(0, k(\cdot,\cdot)\big), \tag{5.43}$$

so that a calibrated PDF is given by

$$g_{\hat{R}|\mathbf{X}}(\hat{r}) = \mathrm{Cauchy}\big(\hat{r}; \vartheta = \mu_{\hat{R}|\mathbf{X}}, \gamma = (w_\gamma(f_{\hat{R}|\mathbf{X}}) \cdot \sigma_{\hat{R}|\mathbf{X}})\big). \tag{5.44}$$

This method is denoted by GP-Cauchy.

5.4.1 Joint Multivariate Calibration

For the task of object detection, it is required to determine the object position and shape which results in a multivariate regression problem. Most approaches for probabilistic object detection use independent normal distributions for each bounding box dimension [71, 40, 70]. Therefore, we further extend the GP recalibration methods GP-Beta, GP-Normal, and GP-Cauchy to jointly infer the calibrated distributions using all available input information. Similar to the position-dependent confidence calibration in Sec. 3.3, the joint recalibration allows to capture possible correlations between the input quantities. For this reason, we adapt the GP recalibration scheme by [41] and use a multi-output GP [115, 116, 117] which estimates the latent functions for each dimension given the uncalibrated input distributions, where L is the number of bounding

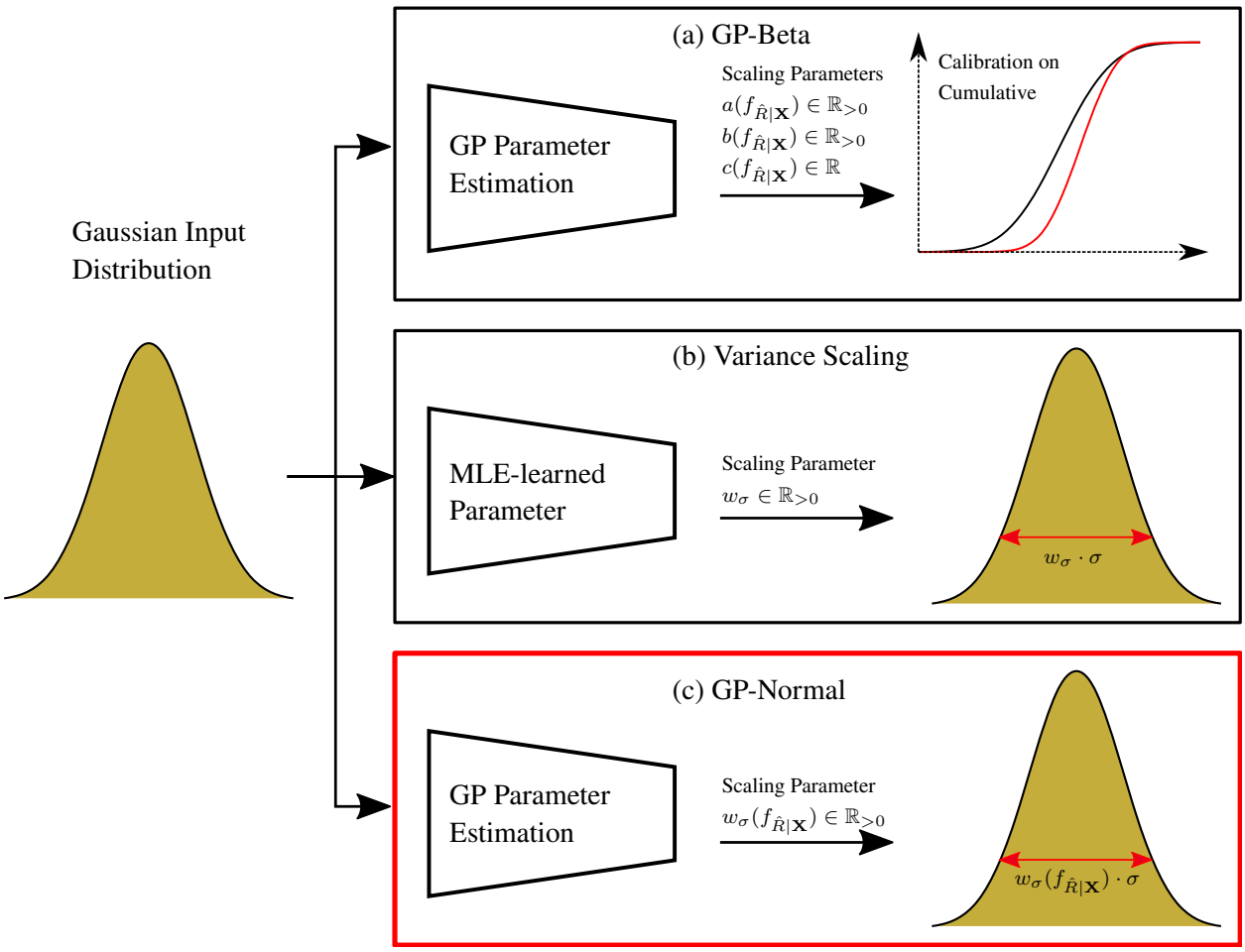


Figure 5.5: Schematic representation of the work principle of various calibration methods and how we derive the GP-Normal method. (a) The GP-Beta [41] is a non-parametric calibration method that uses a Gaussian Process (GP) to flexibly obtain the calibration parameters based on the input distribution. (b) In contrast, the Variance Scaling [42, 18] yields a parametric normal distribution as calibration output but only uses a single scaling parameter for the recalibration of the input variance. (c) Our GP-Normal method also yields a Gaussian as calibration output but uses the flexible GP parameter estimation scheme of the GP-Beta. In this way, it is possible to keep a parametric Gaussian representation of the input distribution as well as to perform input-sensitive uncertainty calibration.

box dimensions. For the GP-Beta method [41], the parameter estimation for the multivariate case extends to

$$\mathbf{w}_a(\cdot), \mathbf{w}_b(\cdot), \mathbf{w}_c(\cdot) \sim \mathrm{gp}\big(0, \mathbf{k}(\cdot,\cdot), \mathbf{B}_\beta\big), \tag{5.45}$$

with scaling functions $\mathbf{w}_a(f_{\hat{\mathbf{R}}|\mathbf{X}}), \mathbf{w}_b(f_{\hat{\mathbf{R}}|\mathbf{X}}), \mathbf{w}_c(f_{\hat{\mathbf{R}}|\mathbf{X}}) \in \mathbb{R}^L$ that use the uncalibrated multivariate distributions as input and return a scaling weight for each bounding box dimension L. In this case, the coregionalization matrix $\mathbf{B}_\beta$ captures the dependencies between all parameters and all dimensions, so that it is given by $\mathbf{B}_\beta \in \mathbb{R}^{3L\times 3L}$. Furthermore, since we now work with a multivariate input with mean vector $\boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}}$ and (diagonal) covariance matrix $\boldsymbol{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}}$, we use the original (multivariate) formulation for the kernel function $\mathbf{k}$

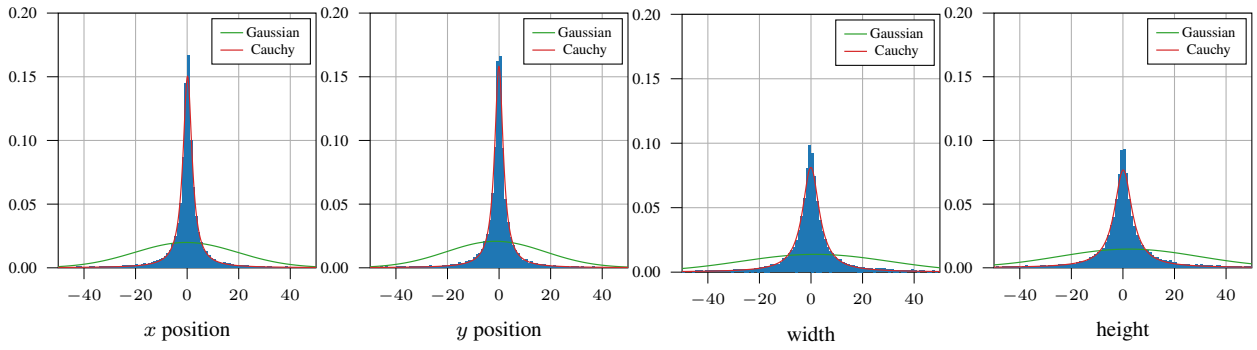


Figure 5.6: Error distribution of the regression output $\hat{\mathbf{R}}$ obtained by a `Faster R-CNN` object detector on the MS COCO data set [6, p. 8, Fig. 3]. The Gaussian fit does not yield a good representation of the error distribution. In contrast, the Cauchy distribution is more suitable as it allows for heavier tails compared to a Gaussian.

[118] which is given by

$$\mathbf{k}\big((\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), (\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)\big) = \theta^L |\boldsymbol{\Sigma}_{ij}|^{-\frac{1}{2}} \exp\Big(-\frac{1}{2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^\top \boldsymbol{\Sigma}_{ij}^{-1}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)\Big), \tag{5.46}$$

where $\boldsymbol{\Sigma}_{ij} = \boldsymbol{\Sigma}_i + \boldsymbol{\Sigma}_j + \theta^2\mathbf{I}$ with identity matrix $\mathbf{I} \in \mathbb{R}^{L\times L}$ [118]. Similarly, the parameter estimation for the GP-Normal in the multivariate case extends to

$$\log\big(\mathbf{w}_{\boldsymbol{\Sigma}}(\cdot)\big) \sim \mathrm{gp}\big(0, \mathbf{k}(\cdot,\cdot), \mathbf{B}_{\mathcal{N}}\big), \tag{5.47}$$

as well as for the GP-Cauchy by

$$\log\big(\mathbf{w}_{\boldsymbol{\gamma}}(\cdot)\big) \sim \mathrm{gp}\big(0, \mathbf{k}(\cdot,\cdot), \mathbf{B}_{\mathrm{C}}\big), \tag{5.48}$$

with $\mathbf{w}_{\boldsymbol{\Sigma}}(f_{\hat{\mathbf{R}}|\mathbf{X}}), \mathbf{w}_{\boldsymbol{\gamma}}(f_{\hat{\mathbf{R}}|\mathbf{X}}) \in \mathbb{R}^L$ as the functions that return a scale weight $w_\sigma^{(l)}$ and $w_\gamma^{(l)}$ for each bounding box dimension $k \in \{1, \ldots, L\}$. Here, the coregionalization matrices are given by $\mathbf{B}_{\mathcal{N}}, \mathbf{B}_{\mathrm{C}} \in \mathbb{R}^{L\times L}$. Although we aim to capture possible correlations of the input for calibration, the resulting calibrated probability distributions are still modeled using independent distributions, so that e.g. the calibrated normal distribution is given by

$$g_{\hat{\mathbf{R}}|\mathbf{X}}(\hat{\mathbf{r}}) = \prod_{l=1}^{L} \mathcal{N}\Big(\hat{r}_l; \mu_{\hat{R}_l|\mathbf{X}}, \big(\mathbf{w}_{\boldsymbol{\Sigma}}(f_{\hat{\mathbf{R}}|\mathbf{X}})^{(l)} \cdot \sigma_{\hat{R}_l|\mathbf{X}}\big)^2\Big). \tag{5.49}$$

This holds for all cases, the non-parametric GP-Beta as well as for the parametric GP-Normal or GP-Cauchy. Thus, the likelihood of the GP for parameter training is simply the product of the likelihood functions for each bounding box dimension. $\big($cf. (5.32)$\big)$. Therefore, it is now possible to jointly infer multiple dimensions using the GP recalibration scheme.

5.4.2 Correlation Estimation and Recalibration

Besides the inference of multiple independent probability distributions, it is also possible to infer parametric probability distributions with correlations between the dimensions using the GP recalibration scheme. This allows for a post-hoc introduction of correlations between independently inferred random variables. In this section, we use the multivariate normal distribution to introduce a correlation estimation scheme. In a first step, we compute the marginal correlation coefficients $\rho_{ij} \in [-1, 1]$ between all dimensions $i, j \in \{1, \dots, L\}$. The correlation coefficients are used to compute the full covariance matrices $\mathbf{\Sigma}_{\hat{\mathbf{R}}|\mathbf{X}}$ with covariances $\sigma_{ij} = \rho_{ij}\sigma_i\sigma_j$ for all samples which can be interpreted as a prior for the calibrated covariances. Afterwards, we use the $\mathbf{LDL}^\top$ decomposition of $\mathbf{\Sigma}$ where $\mathbf{L}$ is a lower triangular matrix and $\mathbf{D}$ the respective diagonal. Using the decomposed representation, we can obtain the input-dependent scale weights $\mathbf{w_L}(f_{\hat{\mathbf{R}}|\mathbf{X}}) \in \mathbb{R}^{L\times L}$ and $\mathbf{w_D}(f_{\hat{\mathbf{R}}|\mathbf{X}}) \in \mathbb{R}^{L\times L}_{>0}$ for the lower triangular and the diagonal matrix, respectively, by

$$\log\big(\mathbf{w_D}(\cdot)\big), \mathbf{w_L}(\cdot) \sim \mathrm{gp}\big(0, \mathbf{k}(\cdot,\cdot), \mathbf{B}_{\mathcal{N}}\big). \tag{5.50}$$

This allows for a reconstruction of a calibrated covariance matrix by

$$\mathbf{\Sigma} = \big(\mathbf{w_L}(f_{\hat{\mathbf{R}}|\mathbf{X}}) \odot \mathbf{L}\big)\big(\mathbf{w_D}(f_{\hat{\mathbf{R}}|\mathbf{X}}) \odot \mathbf{D}\big)\big(\mathbf{w_L}(f_{\hat{\mathbf{R}}|\mathbf{X}}) \odot \mathbf{L}\big)^\top, \tag{5.51}$$

where $\odot$ denotes the element-wise product. The rescaling of a $\mathbf{LDL}^\top$ decomposed matrix is numerically more stable as it preserves a symmetric and positive semidefinit covariance matrix after calibration. The NLL of the multivariate GP is simply obtained by the likelihood of the multivariate normal distribution. Therefore, it is now possible not only to jointly recalibrate multiple dimensions but also to introduce correlations between independently inferred random variables. This method can also be used for a covariance recalibration as well, if the input is modeled with a non-diagonal covariance matrix. In this case, we can omit the first step of computing the marginal correlation coefficients and directly use the provided covariances as the priors for the GP recalibration. Similar to the standard GP-Normal method, covariance estimation and covariance recalibration can be seen as the multivariate extension of the definition for variance calibration towards multivariate distribution calibration using normal distributions, so that

$$\mathbb{E}_{\mathbf{X},\bar{R}}\big[(\bar{\mathbf{R}} - \boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}})(\bar{\mathbf{R}} - \boldsymbol{\mu}_{\hat{\mathbf{R}}|\mathbf{X}})^\top \big| \Pi_{\bar{\mathbf{R}}} = \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})\big] = \mathbf{\Sigma}, \tag{5.52}$$

is fulfilled for all $\boldsymbol{\mu} \in \mathcal{R}$ and all $\mathbf{\Sigma} \in \mathbb{R}^{K\times K}$, where $\mathbf{\Sigma} = \mathbf{\Sigma}^\top$ and $\mathbf{\Sigma} \succeq 0$.

5.5 Experiments for Spatial Uncertainty Calibration

The evaluations presented in this section are part of our publication in [6, pp. 11]. We use a probabilistic `Faster R-CNN` as well as a `RetinaNet` that output Gaussian distributions with mean and variance for each bounding box quantity c_x, c_y, width, and height. The basic architectures for both networks have been adapted from [96] and extended by a probabilistic bounding box regression output (cf. Sec. 2.2.2). We trained both networks on the MS COCO [44] and Berkeley DeepDrive [122] training data sets following the standard training configuration by [96]. Similar to our previous experiments for semantic confidence calibration in Sec. 3.4, we use the predictions on the respective validation data sets for calibration training and evaluation. For this reason, the images of the validation sets are divided randomly which yields equally sized sets with probabilistic predictions for calibration training and evaluation, respectively. We start by examining if the object detection model makes unbiased predictions for the object bounding boxes. The results are shown in Tab. 5.1. For each object detection model, we observe a small bias in the estimated object position. However, compared to the error variance, the bias is in a range where it does not have a large impact on our further evaluations.

For our experiments, we adopt Isotonic Regression [34], Variance Scaling [42, 18], GP-Beta [41], and our new parametric GP-Normal and GP-Cauchy methods. As calibration metrics, we use the NLL, Pinball loss, C-QCE, UCE [18], and ENCE [42]. Each metric is reported as the average over all bounding box quantities. Since the Pinball loss and C-QCE are computed for certain quantiles, we further report a mean Pinball loss and a mean C-QCE for quantile levels from 0.05 to 0.95 with steps of 0.05. Furthermore, the C-QCE, UCE,

Table 5.1: Bias and the respective error standard deviation of the object detection models on the data sets for all bounding box quantities (in pixels). We observe a small bias in each model which, however, is in a range where it does not have a large impact on our uncertainty evaluation results.

	DB	Num. Detections	Quantity	Bias	RMSE
`Faster R-CNN`	BDD	69.775	c_x	0.165	7.637
			c_y	0.190	7.857
			w	-0.300	10.602
			h	-0.439	10.441
	COCO	24.807	c_x	0.604	20.406
			c_y	1.455	20.085
			w	-1.716	30.598
			h	-2.509	27.672
`RetinaNet`	BDD	108.133	c_x	0.351	9.531
			c_y	0.196	8.199
			w	-0.640	12.467
			h	-0.801	11.763
	COCO	50.274	c_x	1.040	26.200
			c_y	3.180	26.622
			w	-1.849	37.102
			h	-5.187	34.999

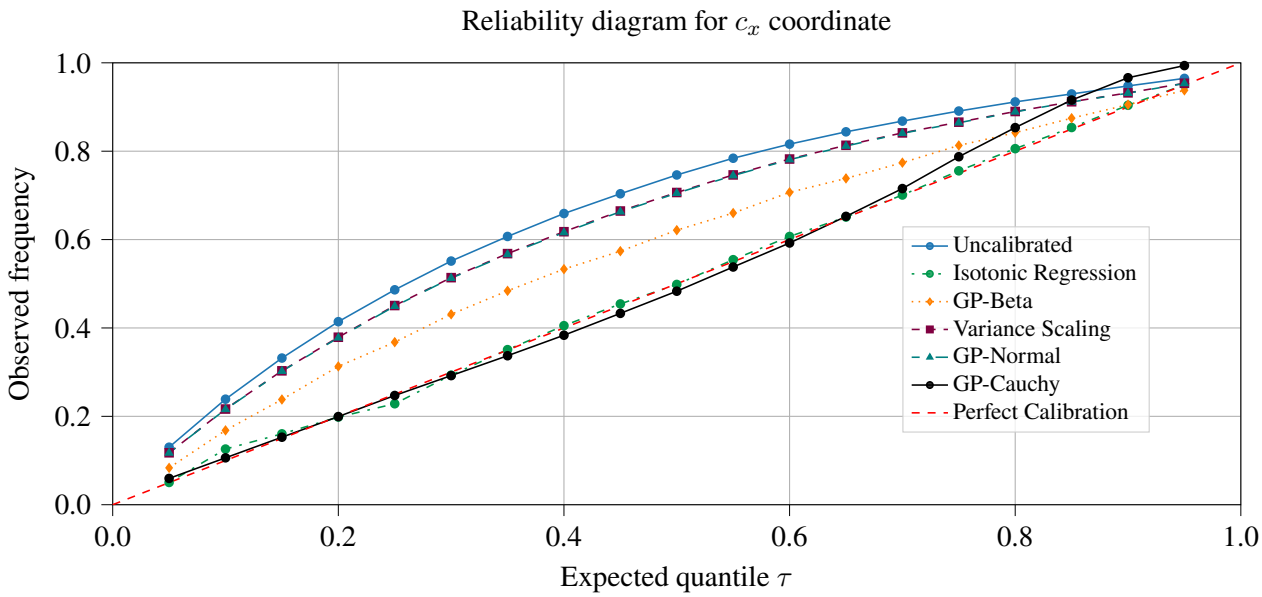


Figure 5.7: Reliability diagram of the regression uncertainty for different quantile levels and for different calibration methods [6, p. 13, Fig. 4]. The expected quantile level (x axis) is compared to the observed prediction interval coverage frequency of the ground-truth scores over all samples (y axis). The diagram shows the miscalibration in terms of quantile calibration for the predicted uncertainty of the c_x coordinates before and after calibration. We can observe an underconfidence of the uncalibrated uncertainty estimates. In this case, Isotonic Regression and GP-Cauchy achieve the best calibration performance.

and ENCE require a binning scheme over the predicted variance/standard deviation, so that we use $I = 20$ bins for binning. Note that the GP-Cauchy calibration method yields a Cauchy distribution which, however, has no statistical moments such as expectation and variance. Thus, we can not compute the UCE and ENCE after calibration with GP-Cauchy.

Subsequently, we use the GP-Normal method to perform covariance estimation (cf. Sec. 5.4.2). This variant is denoted by GP-Normal (mv.). In this way, it is possible to capture possible correlations between the bounding box quantities. For the multivariate evaluation, we also use the NLL and the C-QCE, where the latter one is currently only defined for Gaussian distributions. Except for the multivariate GP-Normal, the NLL is obtained by assuming independent probability distributions for each bounding box quantity. The calibration results for the univariate as well as for the multivariate case are presented in Tab. 5.2. Furthermore, we show the reliability diagram in Fig. 5.7 which allows for an evaluation of the probabilistic forecaster by means of quantile calibration.

The results show that Isotonic Regression calibration [34] achieves the best results for NLL, Pinball loss, and C-QCE. This shows that Isotonic Regression achieves the best results for quantile calibration in our experiments. As already mentioned in Sec. 5.3, the error distributions of the examined object detectors rather follow a Cauchy distribution than a Gaussian. This can be seen in Fig. 5.6. Thus, the GP-Cauchy method is also able to achieve qualitatively good calibration results. This is also underlined by the observations

Table 5.2: Calibration results for the probabilistic bounding boxes of `Faster R-CNN` and `RetinaNet` object detectors before and after regression uncertainty calibration [6, p. 12, Tab. 1]. The best calibration scores are highlighted in bold. In all cases, we observe a miscalibration for the uncalibrated regression uncertainty. Furthermore, Isotonic Regression is able to achieve the best calibration performance in terms of quantile calibration (cf. Pinball loss and C-QCE) compared to the remaining calibration methods. In contrast, Variance Scaling, GP-Normal, and GP-Beta achieve the best results for variance calibration (cf. UCE and ENCE).

Setup			Univariate					Multivariate	
	DB	Method	NLL	$\overline{\text{C-QCE}}$	$\overline{\mathcal{L}_{\text{Pin}}}$	UCE	ENCE	NLL	$\overline{\text{C-QCE}}$
Faster R-CNN	BDD	Uncalibrated	3.053	0.040	1.079	19.683	0.454	12.210	**0.071**
		Isotonic Reg.	**2.895**	**0.017**	**1.059**	39.157	0.303	**11.579**	-
		GP-Beta	2.941	0.057	1.077	3.635	0.199	11.764	-
		Var. Scaling	2.962	0.061	1.086	3.361	**0.175**	11.848	0.131
		GP-Normal	2.962	0.059	1.084	3.289	0.188	11.848	0.128
		GP-Normal (mv.)	2.968	0.054	1.150	**3.234**	0.191	11.584	0.133
		GP-Cauchy	3.011	0.050	1.189	-	-	12.045	-
	COCO	Uncalibrated	3.561	0.154	3.055	32.899	0.096	14.245	0.256
		Isotonic Reg.	**3.340**	**0.020**	**2.715**	42.440	0.121	**13.360**	-
		GP-Beta	3.412	0.074	2.750	51.455	0.140	13.649	-
		Var. Scaling	3.554	0.131	2.952	33.155	0.093	14.216	0.222
		GP-Normal	3.554	0.130	2.949	48.167	0.132	14.235	**0.200**
		GP-Normal (mv.)	3.562	0.121	3.298	**28.382**	**0.087**	13.955	0.216
		GP-Cauchy	3.406	0.039	2.897	-	-	13.624	-
RetinaNet	BDD	Uncalibrated	4.052	0.130	1.847	39.933	0.491	16.208	0.234
		Isotonic Reg.	**3.224**	**0.068**	**1.814**	39.498	0.252	**12.898**	-
		GP-Beta	3.419	0.091	2.169	**14.892**	**0.173**	13.677	-
		Var. Scaling	3.392	0.095	2.203	15.833	0.180	13.568	**0.131**
		GP-Normal	3.392	0.095	2.205	15.820	0.180	13.568	**0.131**
		GP-Normal (mv.)	3.434	0.097	3.356	15.342	0.200	13.353	0.133
		GP-Cauchy	3.316	0.097	1.944	-	-	13.262	-
	COCO	Uncalibrated	4.694	0.115	4.999	553.244	0.530	18.778	**0.153**
		Isotonic Reg.	**3.886**	**0.029**	**4.534**	105.343	0.113	**15.544**	-
		GP-Beta	4.169	0.142	5.093	**77.017**	**0.071**	16.677	-
		Var. Scaling	4.204	0.167	5.606	83.653	0.072	16.815	0.207
		GP-Normal	4.204	0.167	5.606	84.367	0.072	16.815	0.207
		GP-Normal (mv.)	4.236	0.158	6.233	82.074	0.087	16.455	0.194
		GP-Cauchy	3.936	0.043	4.716	-	-	15.745	-

within the reliability diagram in Fig. 5.7. In contrast to the semantic confidence calibration, we observe an underconfidence of the predicted uncertainty estimates. Interestingly, Isotonic Regression also outperforms the non-parametric GP-Beta [41] which is originally designed to achieve distribution calibration. On the one hand, GP-Beta is restricted to the beta calibration family of functions which, however, limits the calibration power of this method. On the other hand, we can not find a strong connection between miscalibration and position/shape information. This can also be seen by the minor differences in the calibration between Isotonic

Regression and GP-Beta, as well as between Variance Scaling [42, 18] and GP-Normal. Both methods, Variance Scaling and GP-Normal, yield parametric Gaussians as calibration output, whereas GP-Normal also has the flexibility to capture possible correlations between miscalibration and position/shape information. However, we only observe minor differences in calibration which leads to the assumption of only minor correlations between position/shape information and miscalibration.

The best calibration results for UCE and ENCE are obtained by Variance Scaling [42, 18] and GP-Normal which perform equally in our experiments. Thus, both methods achieve the best results in terms of variance calibration. In addition, the multivariate GP-Normal is able to further improve the multivariate NLL compared to the (independent) Variance Scaling and GP-Normal recalibration. As described in Sec. 5.2.3, a variance-calibrated forecaster is also quantile-calibrated if the predicted estimates are unbiased and the ground-truth data are normally distributed. However, we already stated that the observed error distributions are not optimally fit by Gaussians so that variance calibration does not lead to optimal quantile calibration in our experiments. Nevertheless, this might not be an issue if subsequent processes such as Kalman filtering assert Gaussian distributions as input. In this case, calibration with Variance Scaling and GP-Normal are advantageous since they are designed to provide an optimal Gaussian fit. The influence of regression uncertainty calibration for a subsequent Kalman filtering is part of the experiments within the next Chap. 6.

Therefore, we conclude that the simple Isotonic Regression [34] achieves the best calibration results and thus is sufficient to achieve quantile-calibrated uncertainty estimates. If the estimated quantiles of a probabilistic object detector are of interest in application, then Isotonic Regression should be the preferred method. In contrast, if a parametric distribution is required, the GP-Cauchy leads to an optimal fit for the observed error distributions in our examinations. Further investigations are necessary to gain more evidence if this also holds for other detection architectures. Finally, the Variance Scaling [42, 18] and (multivariate) GP-Normal lead to the best results regarding variance calibration and thus are able to achieve the best Gaussian fit in our experiments. These methods should be preferred if a Gaussian representation of the spatial uncertainty is required in a subsequent application.

5.6 Conclusion for Spatial Uncertainty Calibration

Since not only semantic class but also spatial position uncertainty is an important part of environment perception, we examine and evaluate the spatial position uncertainty of common probabilistic object detectors in this chapter. For this reason, we give an overview of the recent definitions for uncertainty calibration of regression tasks and set these definitions into a common mathematical context. Thus, we are able to show that a variance-calibrated forecaster [42, 18] is also quantile-calibrated [34] for unbiased predictions and normally distributed data. Furthermore, we show the connection between distribution calibration [41] and variance calibration [42, 18]. Since an object detection model needs to jointly estimate the position and shape information of detected objects, we extend these definitions to the multivariate case.

In the next step, we present the most recent methods for regression uncertainty calibration. In this scope, we

provide detailed descriptions for the non-parametric Isotonic Regression [34] and GP-Beta [41] for quantile calibration and distribution calibration, respectively, as well as for the parametric Variance Scaling [42, 18] for variance calibration. We extend the existing calibration methods and propose the GP-Normal and GP-Cauchy methods [6] which both adapt the approach of distribution calibration using a Gaussian process (GP) for parameter estimation. In contrast to the non-parametric GP-Beta [41], the GP-Normal and GP-Cauchy yield parametric normal and Cauchy distributions as calibration output. This might be advantageous for subsequent applications such as Kalman filtering that require parametric distributions as input. We provide more detailed experiments on the effect of calibration on object tracking in Chap. 6. Furthermore, we use the GP recalibration framework to jointly calibrate all dimensions that are necessary for bounding box inference. This allows to capture possible correlations between position/shape information and miscalibration. We also use the GP-Normal to derive a covariance estimation scheme which allows for a post-hoc introduction of correlations between independently inferred random variables.

Our experiments show that the simple Isotonic Regression [34] is able to achieve the best calibration results in terms of quantile calibration. We can not find a strong connection between position/shape information and miscalibration so that the (marginal) recalibration using Isotonic Regression is sufficient within our experiments. In contrast, the Variance Scaling [42, 18] and GP-Normal achieve the best results for a Gaussian fit of the predicted uncertainty to the observed error distribution which, in turn, leads to the best results for variance calibration. Furthermore, the GP-Normal method also allows for a post-hoc introduction of correlations between independently inferred random variables. This allows for a further improvement in the multivariate calibration case. Therefore, we conclude that Isotonic Regression is the method of choice if the predicted quantiles are of special interest. However, if a parametric normal distribution is required, we recommend to use Variance Scaling [42, 18] or the (multivariate) GP-Normal. This assumption is underlined by our examinations for uncertainty calibration on object tracking which can be found in Chap. 6.

6 Application of Calibration to Object Tracking

In the previous chapters, we focused on the task of object detection whose target is to identify objects within a single frame. In contrast, the task of Multiple Object Tracking (MOT) is to identify the same objects in subsequent frames within a sequence of images. In object tracking, we are interested in the position information over time as well as the belief that the actual track matches a ground-truth object, i.e., if the tracked object exists. In this chapter, we demonstrate how the uncertainty calibration methods for object detection from the previous Chap. 3 and Chap. 5 can be integrated into an object tracking framework. For this reason, we elaborate the mathematical context of object tracking and further show how to include semantic confidence calibration as well as spatial uncertainty calibration in the estimation of the object's existence as well as for its position/shape information, respectively. We do not provide a state-of-the-art tracking framework but rather aim to demonstrate the usefulness of calibration for subsequent applications such as object tracking.

In Sec. 6.1, we start by introducing the basic concept of tracking-by-detection algorithms which utilizes object detectors to generate the observations for the tracking process [123, 124]. Furthermore, we introduce recursive Bayesian filtering in Sec. 6.2 that is the underlying framework for object tracking. Subsequently, we use this concept in Sec. 6.3 to derive a mathematical expression for the estimation of the object existence within a sequence of frames. On the one hand, this expression allows for a direct integration of the frame-wise detector confidence into the estimation of the object's confidence over time. On the other hand, we can show how to directly include the semantic confidence calibration methods into object tracking. Similarly, we introduce the Kalman filter in Sec. 6.4 that is an implementation of recursive filtering for the object position using Gaussian distributions. We show how to include the uncertainty information for the object position provided by the underlying probabilistic object detector. This uncertainty can also be recalibrated using the calibration methods for spatial regression uncertainty. In Sec. 6.5, the basic functionality for the initialization of new tracks as well as the association between existing tracks and new detections is shown so that we have everything together to run our evaluations for object tracking in Sec. 6.6. Finally, we summarize our contributions and findings in Sec. 6.7.

Contributions: In summary, we provide the following contributions to the field of object tracking:

- Derivation of a confidence likelihood to estimate the probability of object existence over time using the detector's confidence.
- Integration of semantic confidence calibration methods into the estimation of object existence.
- Integration of time-varying observation uncertainty for the object position which is provided by probabilistic detection models.
- Integration of spatial uncertainty calibration methods into the estimation of object position.
- Evaluation of the performance and calibration properties of the object tracking when using calibrated uncertainty.

6.1 Object Tracking by Detection

In the scope of object tracking, a relationship between objects detected on different frames over time is established. Common literature on object tracking offers several ways of identifying and tracking single objects in subsequent frames [47, 46, 125]. In this work, we will focus on the tracking-by-detection paradigm [123, 124]. This approach utilizes an object detector to identify multiple objects within a single frame. Subsequently, each detected object is assigned to an object already known by the tracker. If no appropriate object has been found, a new track is generated (cf. Sec. 6.5). In this scope, each track is within a dedicated state that consists of the position/shape information, the object's velocity and/or acceleration, as well as a confidence score that represents the tracker's belief of matching a real ground-truth object.

As already mentioned in the previous chapters, common object detectors provide the position/shape information as well as a confidence for each detected object. The confidence represents the detector's belief of correctness about the individual detection. If we utilize a probabilistic object detector as shown in Chap. 5, it is also possible to obtain uncertainty information for the position/shape information as well. Thus, we can provide the object position/shape as well as the semantic and spatial detection uncertainty to the tracking algorithm. In this chapter, we study the effect of uncertainty calibration on the task of object tracking. For this reason, we adapt our calibration methods for semantic confidence (Chap. 3) and spatial uncertainty calibration (Chap. 5) and use them as an intermediate calibration step before passing the uncertainty information to the tracking algorithm. This concept is schematically shown in Fig. 6.1.

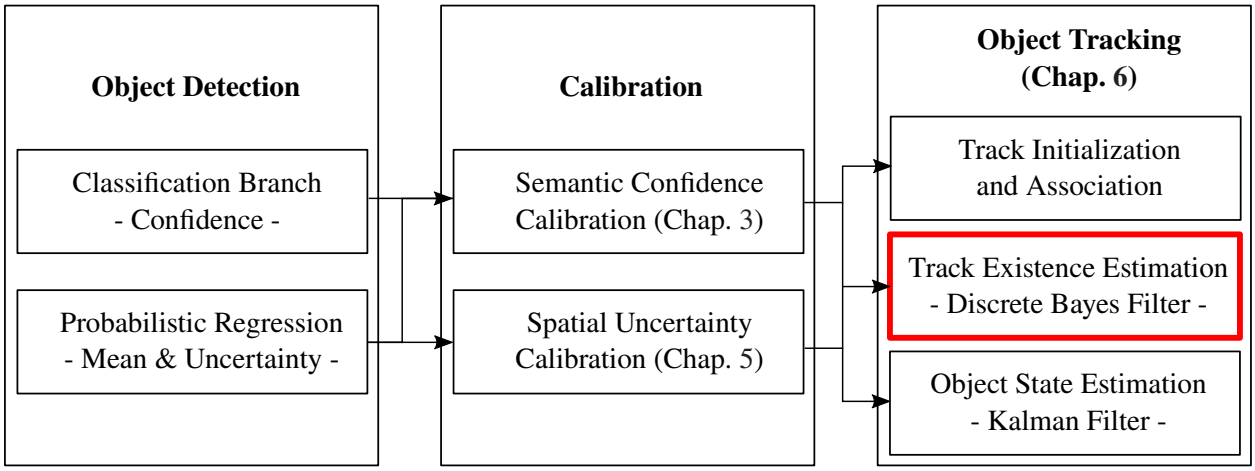


Figure 6.1: Concept of object tracking by detection with additional uncertainty calibration. First, a probabilistic object detector predicts multiple objects in a single frame with uncertainty information for the object existence and position information. This information is used during object tracking to establish a relationship between objects in consecutive frames within a sequence of images. To study the effect of our proposed uncertainty calibration methods from Chap. 3 and Chap. 5, we add an additional uncertainty calibration step between object detection and object tracking.

6.2 Recursive Bayesian Filtering

In image-based object tracking-by-detection, we seek to infer the state of an object (e.g., its position and shape, velocity, acceleration) at time step t given the predictions by an object detection model. In this scope, an object detection model is used to identify objects within a single frame. The detection model outputs a label $\hat{Y}_t \in \mathcal{Y}$ with an according confidence $\hat{P}_t \in [0, 1]$ indicating its belief of matching real-world object which we denote as $\hat{M}_t \sim \text{Bern}(\hat{P}_t)$. Furthermore, the detector outputs a probabilistic (Gaussian) estimate for the object position $\hat{\mathbf{R}}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\hat{\mathbf{R}},t}, \boldsymbol{\Sigma}_{\hat{\mathbf{R}},t})$ with mean $\boldsymbol{\mu}_{\hat{\mathbf{R}},t} \in \mathcal{R}$ and measurement noise covariance matrix $\boldsymbol{\Sigma}_{\hat{\mathbf{R}},t} \in \mathbb{R}^{L\times L}$, where L is the size of the box encoding. Thus, the aggregated observations at time step t are denoted by $\hat{\mathbf{S}}_t \in \mathcal{S}$, where $\hat{\mathbf{S}}_t = (\hat{P}_t, \hat{Y}_t, \hat{\mathbf{R}}_t)^\top$.

A standard detection model does not establish a relationship between objects in consecutive frames. Furthermore, the complete state $\widetilde{\mathbf{C}}_t \in \mathcal{C}$ of an object at time step t is not directly observable given a prediction within a single frame [46, 47, 125]. For example, it is possible to denote the position of an object given the output of the underlying object detector. However, we can not quantify a velocity and an acceleration of an object given a single frame [46, 47, 125] as these quantities require more observations of the same object over time. In our setup, the internal object state for the position information is represented by a Gaussian kinematic state model of second order [46, p. 268 ff.], i.e., the state information does not only contain the actual position/shape but also its velocity (first order derivative) and acceleration (second order derivative). These information are obtained during the tracking process.

In object tracking, we are interested in the (joint) probability distribution for the actual state $\widetilde{\mathbf{c}}_t$ given all previous observations $\hat{\mathbf{s}}_1, \ldots, \hat{\mathbf{s}}_{t-1}$ as well as the actual observation $\hat{\mathbf{s}}_t$, so that the Probability Density Function (PDF) for the actual state is denoted by $f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t|\hat{\mathbf{s}}_1, \ldots, \hat{\mathbf{s}}_t)$. We further use $f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t|\hat{\mathbf{s}}_{1:t})$ for notation simplicity. Under the Markov assumption, that the actual state $\widetilde{\mathbf{c}}_t$ solely depends on the last state $\widetilde{\mathbf{c}}_{t-1}$ as well as on the actual observation $\hat{\mathbf{s}}_t$ [46], the PDF of the updated state variables is given by

$$f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t|\hat{\mathbf{s}}_{1:t}) = \frac{f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t|\widetilde{\mathbf{c}}_t) f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t|\hat{\mathbf{s}}_{1:t-1})}{\int_{\mathcal{C}} f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t|\widetilde{\mathbf{c}}_t^*) f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t^*|\hat{\mathbf{s}}_{1:t-1}) \mathrm{d}\widetilde{\mathbf{c}}_t^*}. \tag{6.1}$$

The term $f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t|\widetilde{\mathbf{c}}_t)$ denotes the likelihood of the observation $\hat{\mathbf{s}}_t$ given the state $\widetilde{\mathbf{c}}_t$, whereas $f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t|\hat{\mathbf{s}}_{1:t-1})$ denotes the probability distribution of the (new) state $\widetilde{\mathbf{c}}_t$ given all previous observations. The latter term can be rewritten using the Chapman–Kolmogorov equation [46], so that it is given by

$$f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t|\hat{\mathbf{s}}_{1:t-1}) = \int_{\mathcal{C}} f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t|\widetilde{\mathbf{c}}_{t-1}) f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_{t-1}|\hat{\mathbf{s}}_{1:t-1}) \mathrm{d}\widetilde{\mathbf{c}}_{t-1}, \tag{6.2}$$

where $f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_t|\widetilde{\mathbf{c}}_{t-1})$ denotes the probability of a transition from the previous state $\widetilde{\mathbf{c}}_{t-1}$ to the actual one $\widetilde{\mathbf{c}}_t$. Furthermore, $f_{\widetilde{\mathbf{C}}}(\widetilde{\mathbf{c}}_{t-1}|\hat{\mathbf{s}}_{1:t-1})$ is nothing else but the state probability distribution of the last time step. Therefore, it is required to provide an appropriate modeling for the observation likelihood function as well as for the state transition distribution. This concept is known as recursive Bayesian filtering. The term in (6.2) is also known as the prediction step as it can be used to generate predictions for the consecutive step without

observing any new data, whereas the equation in (6.1) is known as the update step that updates the predictions for the internal state representations by new incoming observations.

We denote the object state of the estimated position by $\widetilde{\mathbf{R}}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\widetilde{\mathbf{R}},t}, \boldsymbol{\Sigma}_{\widetilde{\mathbf{R}},t})$ with mean vector $\boldsymbol{\mu}_{\widetilde{\mathbf{R}},t} \in \mathbb{R}^{L^*}$ and error covariance matrix $\boldsymbol{\Sigma}_{\widetilde{\mathbf{R}},t} \in \mathbb{R}^{L^* \times L^*}$, where L^* is the size of state's position information, in this case $L^* = 3 \cdot L$ (since the state consists of the actual position/shape, the velocity, and the acceleration). For the state estimation of the object position, a Kalman filter is commonly used which will be explained in Sec. 6.4 in more detail.

Furthermore, we are interested in the belief that the tracked object matches a real ground-truth object given all observations $\hat{\mathbf{S}}_1, \dots, \hat{\mathbf{S}}_t$ until time step t which we denote as $\widetilde{M}_t | \hat{\mathbf{S}}_{1:t} \sim \text{Bern}(\widetilde{P}_t)$ with the track confidence $\widetilde{P}_t \in [0, 1]$. The estimation of the object existence $\widetilde{M}_t$ can be realized using a discrete Bayes filter. Both concepts are a realization of the Bayesian filter framework in (6.1) and (6.2) for continuous and discrete random variables, respectively. We further show how to include the calibration methods for semantic confidence calibration (Chap. 3) and spatial regression uncertainty calibration (Chap. 5) into the tracking environment.

6.3 Discrete Bayes Filter for Object Existence Estimation

For a proper track management, it is necessary to decide when a track no longer exists, e.g., when it has not been recognized correctly or the object has left the image area. Commonly, simple techniques such as exponential moving average (EMA) filters are used to implement a basic track management [126]. However, such techniques are not aware of the uncertainty that is indicated by the underlying detection model as they commonly use the information if an appropriate detection has been found for an existing track or not. In this section, we therefore propose a new track management using the detector confidence for the estimation of the object existence.

As already mentioned in Sec. 3.2.2, the probability of a match $\mathbb{P}(\hat{M}_t = 1)$ for an object within a single frame is a shorthand notation for $\mathbb{P}(\hat{Y}_t = \bar{Y}_t, \hat{\mathbf{R}}_t = \bar{\mathbf{R}}_t)$, where $\hat{Y}_t \in \mathcal{Y}$ and $\hat{\mathbf{R}}_t \in \mathcal{R}$ are the predicted label and position of an object at time step t, respectively. An predicted object is considered to match a real object if the IoU between predicted bounding box and a ground truth box is above a certain Intersection over Union (IoU) threshold (in our case above 0.5). Furthermore, let $\bar{Y}_t \in \mathcal{Y}$ and $\bar{\mathbf{R}}_t \in \mathcal{R}$ denote the respective ground-truth information for the object label and position, respectively. For each detection, the underlying object detector outputs a confidence score $\hat{P}_t$ indicating its belief that the prediction matches a ground-truth object, i.e., for $\hat{M}_t = 1$, so that $\hat{M}_t \sim \text{Bern}(\hat{P}_t)$. When using a position-dependent confidence calibration model h, it is also possible to construct a Bernoulli distribution for $\hat{M}_t$ whose probability parameter depends on the complete model output $\hat{\mathbf{S}}_t = (\hat{P}_t, \hat{Y}_t, \hat{\mathbf{R}}_t)^\top$, so that $\hat{M}_t \sim \text{Bern}\big(h(\hat{\mathbf{S}}_t)\big)$ (cf. (3.20) in Sec. 3.3).

For the estimation of the object existence, we are interested in the probability $\widetilde{P}_t \in [0, 1]$ that a track matches a ground-truth object given all previous confidence, label, and bounding box information, where $\hat{\mathbf{S}}_{1:t} = (\hat{P}_{1:t}, \hat{Y}_{1:t}, \hat{\mathbf{R}}_{1:t})^\top$ denotes the aggregated observations provided by the detection model. Thus, we

can interpret the random variable $\widetilde{M}_t|\hat{\mathbf{S}}_{1:t} \sim \text{Bern}(\widetilde{P}_t)$ for the track confidence similarly. Note that we can interpret $\hat{\mathbf{S}}_{1:t}$ either as the aggregated model output, if we use position-dependent confidence calibration during object existence estimation. Alternatively, it is also possible to assume that $\hat{\mathbf{S}}_{1:t}$ only represents the detector's confidence information, if no calibration or standard calibration methods are used.

From equation (3.3) in Sec. 3.2.1, we know that the confidence $\hat{P}_t$ (or its calibrated variants) can be interpreted as the direct estimation of the probability parameter of the Bernoulli distribution of $\hat{M}_t$. However, we can not directly adapt the detector confidence as a raw estimate for the track confidence $\widetilde{P}_t$ as the underlying random variable $\widetilde{M}_t|\hat{\mathbf{S}}_{1:t}$ is conditioned on all previous observations. Thus, the target now is to update the model's belief of matching a real ground-truth object over time given all previous observations $\hat{\mathbf{S}}_{1:t}$. Since $\widetilde{M}_t$ is a discrete random variable, the probability distribution is given as a Probability Mass Function (PMF), and we can adapt the Bayesian filtering equation (6.1) for the existence estimation to

$$\mathbb{P}(\widetilde{m}_t|\hat{\mathbf{s}}_{1:t}) = \frac{f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t|\widetilde{m}_t)\mathbb{P}(\widetilde{m}_t|\hat{\mathbf{s}}_{1:t-1})}{\sum_{\widetilde{m}_t^* \in \{0,1\}} f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t|\widetilde{m}_t^*)\mathbb{P}(\widetilde{m}_t^*|\hat{\mathbf{s}}_{1:t-1})}, \tag{6.3}$$

with

$$\mathbb{P}(\widetilde{m}_t|\hat{\mathbf{s}}_{1:t-1}) = \sum_{\widetilde{m}_{t-1} \in \{0,1\}} \mathbb{P}(\widetilde{m}_t|\widetilde{m}_{t-1})\mathbb{P}(\widetilde{m}_{t-1}|\hat{\mathbf{s}}_{1:t-1}), \tag{6.4}$$

as the probability for the new state $\widetilde{m}_t$ given the last observations $\hat{\mathbf{s}}_1, \dots \hat{\mathbf{s}}_{t-1}$, and $f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t|\widetilde{m}_t)$ as the likelihood for the observation given the actual state. Here we can see that applying Bayes' theorem under the Markov assumption in (6.3) leads to a likelihood function only for the actual observation. We can rewrite this likelihood to

$$f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t|\widetilde{m}_t) = \frac{\mathbb{P}(\widetilde{m}_t|\hat{\mathbf{s}}_t)f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t)}{\mathbb{P}(\widetilde{m}_t)}. \tag{6.5}$$

Since the probability solely depends on the actual observation, the random variable $\widetilde{M}_t|\hat{\mathbf{S}}_t$ stands for the same intuitive interpretation of matching a ground-truth object given the actual prediction, which is also reflected by the detector output $\hat{M}_t|\hat{\mathbf{S}}_t$. Therefore, we set $\hat{M}_t|\hat{\mathbf{S}}_t = \widetilde{M}_t|\hat{\mathbf{S}}_t$ so that the probability $\mathbb{P}(\hat{m}_t|\hat{\mathbf{s}}_t) = \mathbb{P}(\hat{m}_t|\hat{p}_t, \hat{y}_t, \hat{\mathbf{r}}_t)$ is our calibrated confidence known from equation (3.11) in Sec. 3.2.2. The marginal probability $\mathbb{P}(\hat{m}_t)$ is nothing else but the average precision of the object detector given a certain IoU threshold. If we now plug-in the likelihood equation in (6.3), the final filter equation for the track existence is given by

$$\mathbb{P}(\widetilde{m}_t|\hat{\mathbf{s}}_{1:t}) = \frac{\mathbb{P}(\hat{m}_t|\hat{\mathbf{s}}_t)f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t)\mathbb{P}(\hat{m}_t)^{-1}\mathbb{P}(\widetilde{m}_t|\hat{\mathbf{s}}_{1:t-1})}{\sum_{\hat{m}_t^* \in \{0,1\}} \mathbb{P}(\hat{m}_t^*|\hat{\mathbf{s}}_t)f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t)\mathbb{P}(\hat{m}_t^*)^{-1}\mathbb{P}(\widetilde{m}_t^*|\hat{\mathbf{s}}_{1:t-1})} \tag{6.6}$$

$$= \frac{\mathbb{P}(\hat{m}_t|\hat{\mathbf{s}}_t)\mathbb{P}(\hat{m}_t)^{-1}\mathbb{P}(\widetilde{m}_t|\hat{\mathbf{s}}_{1:t-1})}{\sum_{\hat{m}_t^* \in \{0,1\}} \mathbb{P}(\hat{m}_t^*|\hat{\mathbf{s}}_t)\mathbb{P}(\hat{m}_t^*)^{-1}\mathbb{P}(\widetilde{m}_t^*|\hat{\mathbf{s}}_{1:t-1})}, \tag{6.7}$$

because the prior $f_{\hat{\mathbf{S}}}(\hat{\mathbf{s}}_t)$ for the detector output is canceled out. In the standard case, it would be sufficient to use the detector confidence $\hat{P}_t$ as the Bernoulli parameter for $\hat{M}_t|\hat{\mathbf{S}}_t$. According to equation (3.20) in

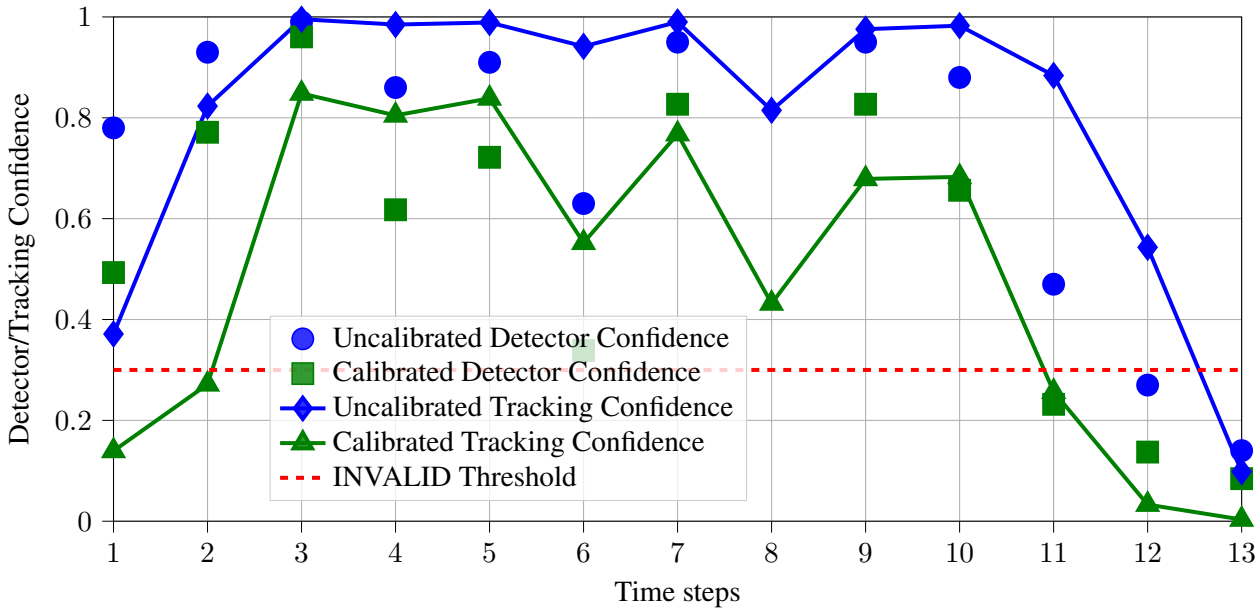


Figure 6.2: Confidence diagram to qualitatively demonstrate the influence of the detector confidence before and after calibration to the estimation of the existence score over the time. During tracking, objects with an existence score below the invalid threshold are discarded.

Sec. 3.3, the distribution for $\hat{M}_t|\hat{\mathbf{S}}_t$ is a Bernoulli using the position-dependent calibration function $h : \mathcal{S} \rightarrow [0,1]$ that maps the detector output $\hat{P}_t, \hat{Y}_t, \hat{\mathbf{R}}_t$ to a calibrated confidence estimate $\hat{Q}_t \in [0,1]$, so that $\hat{M}_t|\hat{\mathbf{S}}_t \sim \text{Bern}\big(h(\hat{P}_t, \hat{Y}_t, \hat{\mathbf{R}}_t)\big)$. In summary, we need to determine the state transition probabilities $\mathbb{P}(\widetilde{m}_t|\widetilde{m}_{t-1})$, the detector precision $\mathbb{P}(\hat{m}_t)$ on a dedicated data set, and a calibration function $h(\cdot)$ to model the track existence over time using our position-dependent confidence calibration methods from Sec. 3.3. Additionally, a confidence threshold is required to decide when to discard an existing track. The influence of the detector confidence on the object existence estimation before and after calibration is qualitatively demonstrated in Fig. 6.2 using the precision and transition probabilities from our experiments in Sec. 6.6.

6.4 Kalman Filter for Object Position Tracking

The Kalman filter is an implementation of the recursive Bayes filter in (6.1) for normally distributed state models and observations. In common applications, the Kalman filter is used to track the position and size of an object within a sequence of images. The random variables for the internal state as well as the ones for the observations are represented by multivariate normal distributions. Since a Gaussian is a conjugate prior to itself, the prediction and update steps in (6.2) and (6.1) can be solved analytically which is advantageous especially for real-time applications. In this section, we give a short overview of the basic Kalman filter concept and propose to use the spatial uncertainty of a probabilistic object detector as a time-varying noise for the observation likelihood. We further show how to use the spatial calibration methods from Chap. 5 to

perform calibration of the time-varying observation noise.

As already mentioned, we use a Gaussian kinematic state model of second order [46, p. 268 ff.] which represents the position/shape information as well as the velocity and acceleration of the object's position/shape as Gaussian random variables for the state representation. The state for the position/shape follows a multivariate normal distribution so that $\widetilde{\mathbf{R}}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\widetilde{\mathbf{R}},t}, \boldsymbol{\Sigma}_{\widetilde{\mathbf{R}},t})$ with mean $\boldsymbol{\mu}_{\widetilde{\mathbf{R}},t} \in \mathbb{R}^{L^*}$ and error covariance $\boldsymbol{\Sigma}_{\widetilde{\mathbf{R}},t} \in \mathbb{R}^{L^* \times L^*}$, where L^* is the state size. The observations for each time step are the predicted bounding boxes $\hat{\mathbf{R}}_t \sim \mathcal{N}(\boldsymbol{\mu}_{\hat{\mathbf{R}},t}, \boldsymbol{\Sigma}_{\hat{\mathbf{R}},t})$ obtained by a probabilistic object detector with mean $\boldsymbol{\mu}_{\hat{\mathbf{R}},t} \in \mathcal{R}$ and estimated covariance $\boldsymbol{\Sigma}_{\hat{\mathbf{R}},t} \in \mathbb{R}^{L \times L}$, where L is the size of the box encoding.

As we can see in (6.1), we need to define appropriate functions for the state transition and the observation likelihood. In the setting of a Kalman filter, the state transition is a linear function with transition matrix $\mathbf{F}_t \in \mathbb{R}^{L^* \times L^*}$ that defines the transition from $\widetilde{\mathbf{r}}_{t-1}$ to $\widetilde{\mathbf{r}}_t$. The state transition matrix is used to specify the relationship between position/shape information and the respective velocities/accelerations and is further used to generate predictions for consecutive time steps. To construct a multivariate normal distribution for the state, we seek to introduce Gaussian noise with zero mean and covariance $\boldsymbol{\Psi}_t \in \mathbb{R}^{L^* \times L^*}$ that is also known as the system noise of the Kalman filter. Thus, the state transition model is given by

$$\widetilde{\mathbf{r}}_t = \mathbf{F}_t \widetilde{\mathbf{r}}_{t-1} + \boldsymbol{\varepsilon}_t, \tag{6.8}$$

$$\text{where } \boldsymbol{\varepsilon}_t \sim \mathcal{N}(0, \boldsymbol{\Psi}_t). \tag{6.9}$$

This yields a multivariate normal distribution for the state transition distribution given by

$$f_{\widetilde{\mathbf{R}}}(\widetilde{\mathbf{r}}_t | \widetilde{\mathbf{r}}_{t-1}) = \mathcal{N}(\widetilde{\mathbf{r}}_t; \mathbf{F}_t \boldsymbol{\mu}_{\widetilde{\mathbf{R}},t-1}, \boldsymbol{\Psi}_t) \tag{6.10}$$

If the initial prior belief for $f_{\widetilde{\mathbf{R}}}(\widetilde{\mathbf{r}}_0)$ is also normally distributed, the posterior $f_{\widetilde{\mathbf{R}}}(\widetilde{\mathbf{r}}_t | \hat{\mathbf{r}}_0, \ldots, \hat{\mathbf{r}}_t)$ is always normally distributed as well.

Furthermore, the likelihood is also defined by a linear function with transition matrix $\mathbf{H}_t \in \mathbb{R}^{L \times L^*}$ that translates from the state space to the observation space. Similar to the state transition, a Gaussian noise with zero mean and covariance $\boldsymbol{\Lambda}_t \in \mathbb{R}^{L \times L}$ is added that represents the observation noise within the Kalman filter, so that

$$\hat{\mathbf{r}}_t = \mathbf{H}_t \widetilde{\mathbf{r}}_t + \boldsymbol{\varrho}_t, \tag{6.11}$$

$$\text{where } \boldsymbol{\varrho}_t \sim \mathcal{N}(0, \boldsymbol{\Lambda}_t). \tag{6.12}$$

In this case, the density function for the observation likelihood is a Gaussian of the form

$$f_{\hat{\mathbf{R}}}(\hat{\mathbf{r}}_t | \widetilde{\mathbf{r}}_t) = \mathcal{N}(\hat{\mathbf{r}}_t; \mathbf{H}_t \boldsymbol{\mu}_{\widetilde{\mathbf{R}},t}, \boldsymbol{\Lambda}_t). \tag{6.13}$$

For common object detectors, which only provide deterministic predictions for the object's position and

shape, the covariance matrix $\mathbf{\Lambda}_t$ of the observation noise is fixed. The variance for each random variable in the observation space is commonly set to the mean squared error of the object detector that is obtained on a dedicated training set [125, p. 16].

Instead, if we use a probabilistic object detector, we can directly provide the estimated uncertainty $\mathbf{\Sigma}_{\hat{\mathbf{R}}_t}$ to the filter, so that the observation noise is time-varying. Therefore, we set $\mathbf{\Lambda}_t = \mathbf{\Sigma}_{\hat{\mathbf{R}}_t}$ if our predictions are obtained by a probabilistic object detector. At this point, we can now integrate our regression calibration methods from Chap. 5 to recalibrate the spatial uncertainty in each frame before the application of the filter update. Let $h(\cdot)$ denote a regression calibration method that takes an uncalibrated (multivariate) Gaussian as input and outputs a recalibrated (multivariate) Gaussian distribution as well. The observation noise then changes to $\mathbf{\Lambda}_t = h(\mathbf{\Sigma}_{\hat{\mathbf{R}}_t})$. Note that for the non-parametric uncertainty calibration methods such as Isotonic Regression (Sec. 5.3.1) and GP-Beta (Sec. 5.3.1), a moment-matching from the non-parametric distributions to a normal distribution is necessary to integrate the recalibrated uncertainty into object tracking using Kalman filtering. In contrast, the output of the parametric Variance Scaling (Sec. 5.3.2) and GP-Normal (Sec. 5.4) can be directly used as these methods already return a Gaussian representation of the uncertainty. Furthermore, we directly use the multivariate extension of the GP-Normal (Sec. 5.4.1) for recalibration. Since the probabilistic object detector only estimates the position/shape information independently, we further integrate our covariance estimation scheme presented in Sec. 5.4.2 to include possible correlations between the observation quantities into the observation noise.

In summary, for Kalman filter implementation, the definition of the transition model $\mathbf{F}_t$ is given by the used kinematic state model, whereas the observation matrix $\mathbf{H}_t$ defines the translation from the state space to the observation space. For the observation noise $\mathbf{\Lambda}_t$, we adapt the Gaussian uncertainty of the underlying probabilistic object detector as our observation uncertainty. In this way, we are able to integrate our spatial uncertainty calibration methods from Chap. 5 into object tracking.

6.5 Track Initialization and Association

At each time step t, it is necessary to either assign an observation by the object detector to an existing track or to generate a new track if no appropriate track has been found. The assignment is commonly performed by calculating a distance metric between each observation and each track [46, 125]. As a distance metric, the Normalized Innovations Squared (NIS) is commonly used which is related to the Normalized Estimation Error Squared (NEES) (cf. Sec. 5.2.1) [46, 125]. The NIS is the squared Mahalanobis distance between a prediction by the object detector and the predicted position of an existing track for the actual time step t

$$\text{NIS} := \left(\mu_{\hat{\mathbf{R}}_t} - \mathbf{H}_t \mu_{\widetilde{\mathbf{R}}_t}\right)^\top \left(\mathbf{H}_t \mathbf{\Sigma}_{\widetilde{\mathbf{R}}_t} \mathbf{H}_t^\top + \mathbf{\Sigma}_{\hat{\mathbf{R}}_t}\right)^{-1} \left(\mu_{\hat{\mathbf{R}}_t} - \mathbf{H}_t \mu_{\widetilde{\mathbf{R}}_t}\right), \tag{6.14}$$

where $\mu_{\hat{\mathbf{R}}_t}$, $\mu_{\widetilde{\mathbf{R}}_t}$ are the mean and $\mathbf{\Sigma}_{\widetilde{\mathbf{R}}_t}$, $\mathbf{\Sigma}_{\hat{\mathbf{R}}_t}$ are the predicted track mean and covariance by the object tracker, and the observation mean and covariance by the object detector, respectively. Similar to the NEES (cf. Sec. 5.2.1), the NIS can be interpreted as an equation for an ellipsoid that - given a certain quantile level

Figure 6.3: Concept of Kalman filtering illustrated by the example of a single vehicle. On the first frame (left), an object detector (red) detects a vehicle with a certain spatial uncertainty for the bounding box edges. However, a track has not been initialized so far. In the consecutive frame (right), the object detector still recognizes the object and produces a prediction. In the meantime, the object tracker initialized a track and predicts its next position (green). Finally, it associates the detector observation to the actual track and updates the internal position (blue).

$\tau \in [0, 1]$ - spans a Highest Posterior Density Region (HPDR) for a certain τ and for the aggregated state and observation uncertainty.

Thus, we can construct a distance matrix $\mathbf{D}_{\text{NIS}} \in \mathbb{R}_{>0}^{L^* \times L}$ containing the NIS scores between all existing tracks and all observations. Commonly, a quantile threshold is set for the NIS using a χ_L^2 distribution with L degrees of freedom that determines the region in which an appropriate detection is searched for the assignment. This threshold is used to mark single entries as invalid for the final assignment. For the final association step, we use the Hungarian method [127] which is a combinatorial optimization algorithm that is used to obtain the optimal observation-to-track assignment by minimizing the overall assignment costs given a cost matrix which is the matrix $\mathbf{D}_{\text{NIS}}$ in our case. If no appropriate existing track can be found for an observation, a new track is initialized at the position of the observation. The concept of Kalman filtering is qualitatively shown in Fig. 6.3

6.6 Experiments for Calibration in Object Tracking

To evaluate the influence of calibration on the task of object tracking, we use the Berkeley DeepDrive data set as well as the same `Faster R-CNN` network architecture with a probabilistic regression branch for the object position as within our evaluations for spatial uncertainty calibration (cf. Sec. 5.5). For object tracking evaluation, the Berkeley DeepDrive tracking data set provides 200 sequences with approx. 200 frames per sequence on average with a frame rate of 5 Hz. We split these sequences and use the first half as a training set to train the calibration methods as well as to get the average precision $\mathbb{P}(\hat{m}_t)$ and the state transition probabilities $\mathbb{P}(\hat{m}_t|\hat{m}_{t-1})$ which are both required to set up the object tracking model. For precision calculation and calibration training, we use an IoU threshold of 0.5.

As described in Sec. 6.5, the NIS score between existing tracks and incoming detections in conjunction with the Hungarian algorithm [127] is used for a track-to-detection association. We further use a HPDR with a quantile level of 0.95 to search for appropriate observations for each track. During object tracking, each track with an existence probability below 0.3 is dropped. For object tracking evaluation, we use the MOT metrics accuracy MOTA, precision MOTP, and the ratio of correctly identified detections over the average number of ground-truths and detections which is denoted by IDF1 [128, 129]. The MOTA metric reflects the error ratios of false positives, false negatives, and identity switches and is given by

$$\text{MOTA} := 1 - \left[\frac{\sum_{t=1}^{T} \left[\text{FP}_t + \text{FN}_t + \text{IDSw}_t \right]}{\sum_{t=1}^{T} N_t} \right], \tag{6.15}$$

with T as the number of frames, where N_t is the number of ground-truth samples present at time step t. Furthermore, FP_t, FN_t, and IDSw_t are the false positives, false negatives, and identity switches at time step t, respectively. The MOTP metric denotes the misalignment between the tracks and the respective ground-truth objects and is given by

$$\text{MOTP} := \frac{\sum_{t=1}^{T} \sum_{n=1}^{N} \left[\mathbb{1}(\hat{m}_{t,n} = 1) \cdot ||\widetilde{\mathbf{r}}_{t,n} - \bar{\mathbf{r}}_{t,n}||_2 \right]}{\sum_{t=1}^{T} \text{TP}_t}, \tag{6.16}$$

where TP_t are the number of true positives at time step t and $||\widetilde{\mathbf{r}}_{t,n} - \bar{\mathbf{r}}_{t,n}||_2$ is the Euclidean distance between the estimated track position $\widetilde{\mathbf{r}}_{t,n}$ and the ground-truth position $\bar{\mathbf{r}}_{t,n}$ for object n at time step t, respectively. Besides these metrics, we further calculate the average false positives (FP) and average false negatives (FN) per frame as well as the average ID switches (IDSw) per object. Finally, we denote the fraction of objects whose trajectories have been covered more than 80% (mostly tracked MT), whose trajectories have been covered between 20% and 80% (partially tracked PT), and which are covered only below 20% (mostly lost ML). For the final evaluation of the MOT metrics, we filter all tracks that have an existence probability above 0.5. The tracks with a confidence above the invalid threshold of 0.3 and below the evaluation threshold of 0.5 are used during object tracking but are discarded during MOT evaluation.

In contrast, we evaluate the calibration metrics using all available information to measure the calibration properties on the complete confidence range. Thus, we only apply the invalid threshold of 0.3 only during object tracking. Similar to the evaluations for semantic confidence calibration in Sec. 3.4, we use the Expected Calibration Error (ECE), the position-dependent Detection Expected Calibration Error (D-ECE), the Brier score, and the Negative Log Likelihood (NLL) with the same setup as within Sec. 3.4 to evaluate the calibration properties of the object existence score. For the evaluation of the spatial uncertainty in object tracking, common literature uses the NEES as an evaluation metric [46, 47]. In Sec. 5.2.1, we use the NEES to construct the Marginal Quantile Calibration Error (M-QCE) metric which is more interpretable as it denotes the average error (in percent) between predicted and observed quantile. Therefore, we use the M-QCE as well as the NLL with the same setup as within Sec. 5.5 to evaluate the spatial uncertainty calibration. Note that the evaluation of the NEES is commonly performed on the estimated states directly which requires ground-truth information for the whole state vector. However, no ground-truth information for the velocity

and acceleration are available by the used data set. Furthermore, we perform the state estimation in image coordinates in our setup. To mitigate these limitations, we transform the estimated states to the observation space using (6.11) and evaluate the state estimation using the available ground-truth information for the object location and size in image coordinates. In the next section, we present our results using uncertainty calibration for object tracking.

6.6.1 Evaluations for intermediate Semantic Confidence Calibration

We start with including the semantic confidence calibration methods into the object tracking that have been presented in Sec. 3.3. For confidence calibration, we use the standard Histogram Binning [36], Logistic Calibration [35], and Beta Calibration [43] as well as their multivariate and position-dependent counter parts presented in Sec. 3.3.1 and 3.3.2, respectively. In the following, the position-dependent methods are denoted by "mv." to distinguish between the standard confidence-only and the multivariate calibration methods. For the position-dependent Logistic Calibration and Beta Calibration, we further distinguish between the conditional independent (indep.) and the conditional dependent (dep.) variants which also model correlations between the position information (cf. Sec. 3.3.2). The results of the object tracking with semantic confidence calibration are presented in Tab. 6.1 and Tab. 6.2 with the MOT and calibration metrics, respectively. Furthermore, the calibration results are presented in Fig. 6.4 and 6.5 as reliability diagrams and as a visualization of the track coverage, respectively.

Although we observe a coverage drop in the track trajectories after calibration (cf. MT, PT, and ML in Fig. 6.5), we are able to achieve a significant improvement in the overall MOTA tracking accuracy, the MOTP tracking precision and the IDF1 score. All calibration methods lead to significant improvements in the overall tracking performance, whereas the position-dependent Histogram Binning is able to achieve the best results. We observe a major decrease in the average scores of false positives per frame (FP) but also an increasing score of average false negatives (FN) which, however, is not as large as the drop of false

Table 6.1: Results in the MOT metrics for object existence estimation using semantic confidence calibration The best scores are highlighted in bold. The multivariate Histogram Binning as well as the multivariate and conditionally dependent scaling methods achieve the best results.

Calibration method	Calibration type	MOTA	MOTP	IDF1	IDSw	FP	FN	MT	PT	ML
Uncalibrated	-	36.845	82.679	47.88	2.23	3.06	**3.16**	**0.382**	0.438	**0.180**
Histogram Binning	conf. only	44.201	84.212	50.11	1.68	1.52	4.12	0.273	0.456	0.270
	mv.	**47.404**	83.805	**50.98**	1.64	**1.41**	3.88	0.266	0.497	0.237
Logistic Calibration	conf. only	43.581	84.342	50.06	1.61	1.52	4.22	0.263	0.458	0.279
	mv. (indep.)	42.086	84.396	49.53	1.71	1.64	4.23	0.261	0.458	0.281
	mv. (dep.)	43.226	**84.422**	49.97	**1.58**	1.55	4.25	0.260	0.461	0.279
Beta Calibration	conf. only	43.505	84.371	50.10	1.59	1.53	4.24	0.259	0.459	0.282
	mv. (indep.)	42.941	84.329	49.71	1.67	1.59	4.20	0.270	0.450	0.280
	mv. (dep.)	43.661	84.323	49.87	**1.58**	1.57	4.17	0.267	0.456	0.277

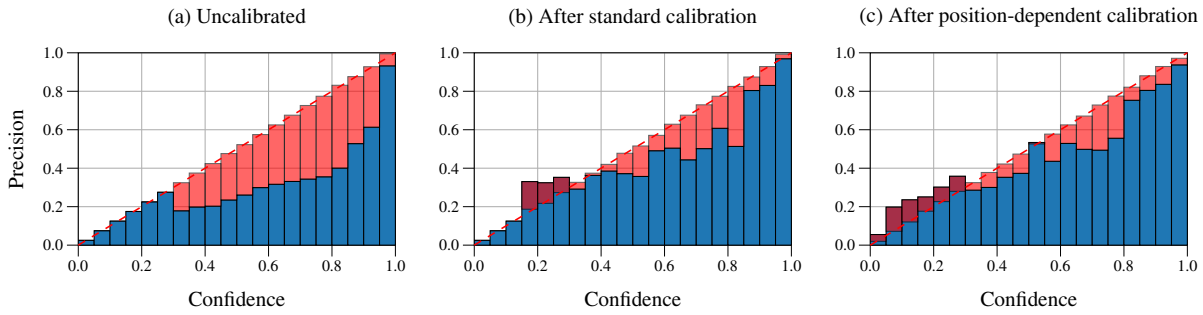
positives. In safety-critical applications, the false negatives are particularly important since objects that are not detected pose a potential safety risk. Changing the confidence threshold value during object tracking for filtering the objects could counteract this phenomenon in order to increase the sensitivity of the model. Furthermore, we observe that the average number of ID switches per object is decreasing (IDSw). If we further inspect the calibration results given in Tab. 6.2, we can see an overall improvement of the confidence calibration properties with intermediate calibration. In this case, the standard confidence methods but also our multivariate and conditionally dependent Logistic Calibration and Beta Calibration (cf. Sec. 3.3.2) are able to achieve the best calibration performance. In Fig. 6.4, we show the reliability diagrams for the track confidence before and after calibration by a standard Histogram Binning [36] as well as by our position-dependent Histogram Binning (cf. 3.3.1). By examining these diagrams, we can see the benefit of position-dependent calibration as it further improves the calibration properties of the object tracking scores. Especially in Fig. 6.4ii, we observe that the position-dependent Histogram Binning leads to an improvement in the calibrated tracking scores across all image regions.

6.6.2 Evaluations for intermediate Spatial Uncertainty Calibration

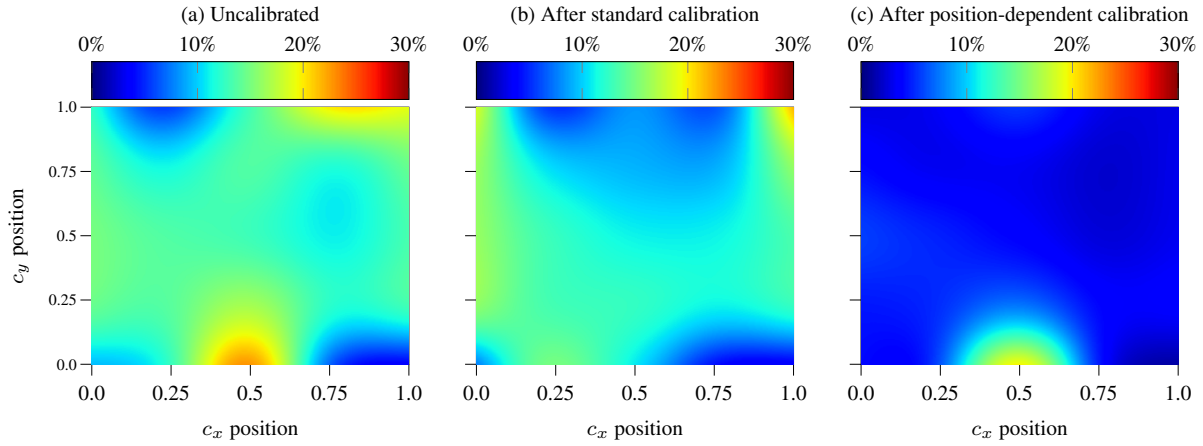
Subsequently, we investigate the effect of spatial uncertainty calibration on object tracking. For this reason, we use the Isotonic Regression [34], Variance Scaling [42, 18], and GP-Beta [41] for uncertainty recalibration. Since Isotonic Regression and GP-Beta yield non-parametric distributions as calibration output, it is necessary to extract the expectation and the variance of the recalibrated distributions which allows for a Gaussian approximation (moment-matching). This is mandatory as the Kalman filter only works with normal distributions. In addition to these methods, we also adapt our GP-Normal uncertainty calibration method presented in Sec. 5.4 which directly yields Gaussian distributions after calibration. In our experiments, we use the GP-Normal under the assumption of independent output variables as well as the multivariate (mv.) GP-Normal that performs covariance estimation of independently learned quantities (cf. Sec. 5.4.2). In this way, we can also evaluate the effect of covariance estimation calibration. The results of the object tracking

Table 6.2: Results in the calibration metrics for object existence estimation using semantic confidence calibration. The multivariate dependent Beta Calibration offers the best calibration performance.

Calibration method	Calibration type	Semantic metrics				Spatial metrics	
		ECE	D-ECE	Brier	NLL	M-QCE	NLL
Uncalibrated	-	0.174	0.174	0.177	0.560	**0.107**	19.095
Histogram Binning	conf. only	0.075	0.080	0.130	0.403	0.117	18.027
	mv.	**0.070**	**0.077**	0.144	0.456	0.119	**15.227**
Logistic Calibration	conf. only	0.073	0.078	0.130	0.390	0.117	19.153
	mv. (indep.)	0.083	0.088	0.134	0.444	0.116	19.870
	mv. (dep.)	0.071	**0.077**	0.128	0.384	0.117	19.146
Beta Calibration	conf. only	0.073	0.079	0.127	0.384	0.117	19.286
	mv. (indep.)	0.080	0.085	0.130	0.424	0.116	19.010
	mv. (dep.)	0.079	**0.077**	**0.114**	**0.357**	0.117	19.239

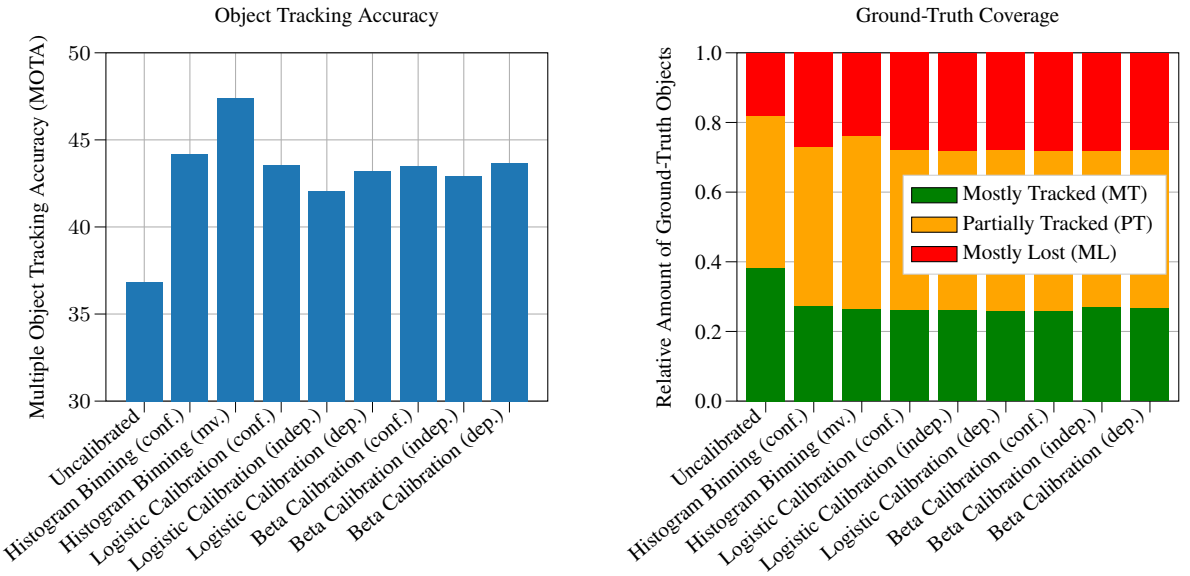


(i) Reliability diagrams w.r.t. the confidence only (0d). The uncalibrated object tracking is consistently overconfident for all confidence levels. This is mitigated by standard Histogram Binning [36] and even further improved using our position-dependent Histogram Binning (cf. Sec. 3.3.1).



(ii) Reliability diagrams w.r.t. the c_x and c_x position of the tracked objects (2d). The standard Histogram Binning [36] improves the calibration properties of the uncalibrated object tracking. However, our position-dependent Histogram Binning is able to reduce miscalibration consistently for nearly all image regions.

Figure 6.4: Reliability diagrams (object tracking) of the semantic confidence for a `Faster R-CNN` on the Berkeley DeepDrive tracking calibration validation set before and after calibration by Histogram Binning. For each frame, we determine for each track if it matches a real ground-truth annotation. In combination with the track confidence for each object at a certain time step, we are able to compute a reliability diagram over all objects and time steps. As already known from the evaluations for object detection calibration in Sec. 3.4.1, the `Faster R-CNN` is consistently too overconfident in its predictions. Accordingly, we can observe the same phenomenon for the track confidence. If we now apply intermediate confidence calibration by standard Histogram Binning, we can also observe a better calibration for the track confidence. The calibration properties are further improved by our position-dependent Histogram Binning (cf. Sec. 3.3.1).



(i) Tracking accuracy before and after calibration.

(ii) Ground-truth coverage before and after calibration.

Figure 6.5: Visualization of the MOT metrics before and after semantic confidence calibration. In this case, the position-dependent (mv.) Histogram Binning achieves the best results in the MOTA metric (left diagram). Furthermore, it is almost able to preserve the tracking coverage of the uncalibrated baseline model (right diagram).

with spatial uncertainty calibration are presented in Tab. 6.3 and Tab. 6.4 with the MOT and calibration metrics, respectively. Furthermore, we present the reliability diagrams as well as a visualization of the trajectory coverage in Fig. 6.6 and 6.7, respectively.

As already suggested in the evaluation of the regression uncertainty calibration methods in Sec. 5.5 and 5.6, the parametric calibration methods Variance Scaling and GP-Normal, that directly output normal distributions as calibration output, offer the best performance in our examinations. Especially the (independent) GP-Normal is able to achieve a significant improvement of the overall tracking accuracy (MOTA). Further-

Table 6.3: Results in the MOT metrics of object tracking using spatial uncertainty calibration of the position uncertainty. The best scores are highlighted in bold. Our (independent) GP-Normal method is able to achieve the overall best tracking accuracy MOTA. The non-parametric Isotonic Regression [34] is able to achieve the best results regarding the ID switches.

Calibration method	MOTA	MOTP	IDF1	IDSw	FP	FN	MT	PT	ML
Uncalibrated	36.845	82.679	47.88	2.23	3.06	3.16	0.382	0.438	0.180
Isotonic Regression	37.315	81.777	**49.76**	**2.00**	3.05	3.18	0.383	0.434	0.183
Variance Scaling	37.557	82.611	48.30	2.18	**3.03**	**3.13**	0.391	0.432	**0.178**
GP-Beta	37.316	82.629	48.30	2.20	3.04	**3.13**	0.388	0.434	**0.178**
GP-Normal	**37.702**	82.608	48.43	2.21	3.04	**3.13**	**0.392**	0.429	0.179
GP-Normal (mv.)	34.917	**82.759**	46.20	2.38	3.18	3.19	0.373	0.446	0.181

more, the (independent) GP-Normal as well as the Variance Scaling are able to reduce the number of false positives while not degrading the amount of false negatives. Interestingly, the multivariate GP-Normal is able to improve the tracking precision (MOTP) and seems to provide very good results for the calibration properties of the estimated quantiles regarding the spatial uncertainty (cf. Fig. 6.6). However, the multivariate GP-Normal also leads to a loss in the tracking accuracy. This is mainly caused by the deterioration in the tracking IDF1 score. We further discuss these observations in the next section.

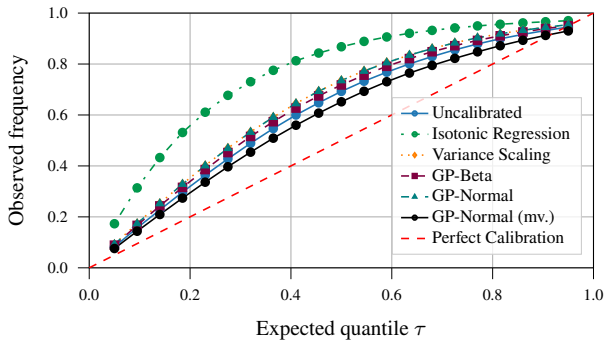
6.6.3 Discussion of the Effect of intermediate Calibration

In our experiments, both types of uncertainty calibration offer an improvement in the tracking performance. Especially in the case of semantic confidence calibration, we assume that the calibrated confidence leads to a significant improvement of the track management. In our evaluations for the MOT metrics, we filter all tracks that have an existence confidence of above 0.5. After semantic confidence calibration, this probability of object existence is more interpretable as it better reflects the true (observed) probability of existence. Thus, we have a higher chance of accessing all tracks that have a true (observed) probability of being alive using a threshold of 0.5. However, we also observer an increase of false negatives which poses a potential safety risk that needs further investigation. Nevertheless, we conclude that an improved confidence score leads to a better object tracking performance. We observe that our position-dependent confidence calibration methods and especially the multivariate Histogram Binning (cf. Sec. 3.3.1) are able to further improve the tracking performance compared to the standard confidence-only calibration methods. Thus, we assume that the additional position-dependency is advantageous to reliably reflect the tracking existence score which, in turn, leads to an improved track management/filtering during object tracking and evaluation.

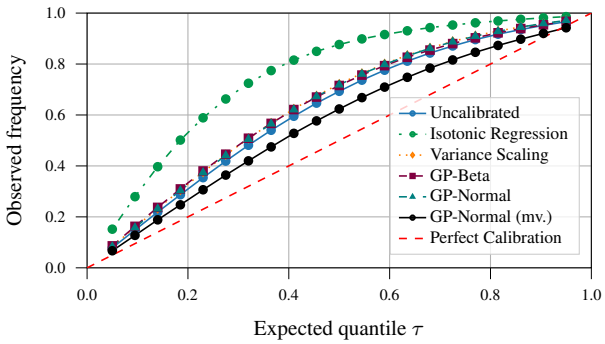
The evaluation of the methods for spatial uncertainty calibration reveals that the non-parametric Isotonic Regression [34] achieves the best results regarding the ID switches and the spatial NLL. An advantage of the non-parametric calibration methods is that they can not only recalibrate the variance but also the mean of the calibrated distribution. Furthermore, the Isotonic Regression has much more degrees of freedom as it offers

Table 6.4: Results in the calibration metrics of object tracking using spatial uncertainty calibration of the position uncertainty. The best scores are highlighted in bold. The GB-Beta is the only calibration method that leads to slight improvements in the metrics for semantic confidence calibration. In contrast, the non-parametric Isotonic Regression [34] leads to a significant improvement in the spatial NLL score. The multivariate GP-Normal achieves the best multivariate quantile coverage.

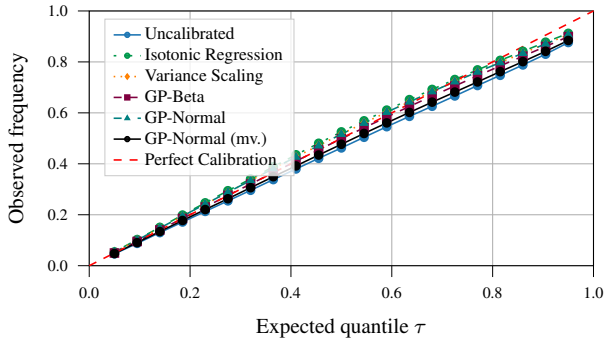
Calibration method	Semantic metrics				Spatial metrics	
	ECE	D-ECE	Brier	NLL	M-QCE	NLL
Uncalibrated	0.174	0.174	0.177	**0.560**	0.107	19.095
Isotonic Regression	0.188	0.188	0.186	0.605	0.204	**15.561**
Variance Scaling	0.175	0.175	0.177	0.562	0.152	17.271
GP-Beta	**0.173**	**0.173**	**0.176**	**0.560**	0.135	17.916
GP-Normal	0.175	0.175	0.177	0.563	0.151	17.279
GP-Normal (mv.)	0.176	0.176	0.179	0.564	**0.073**	21.894



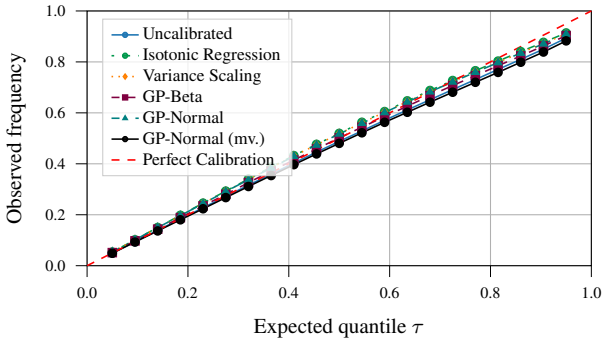
(i) Reliability diagram for the track c_x coordinate.



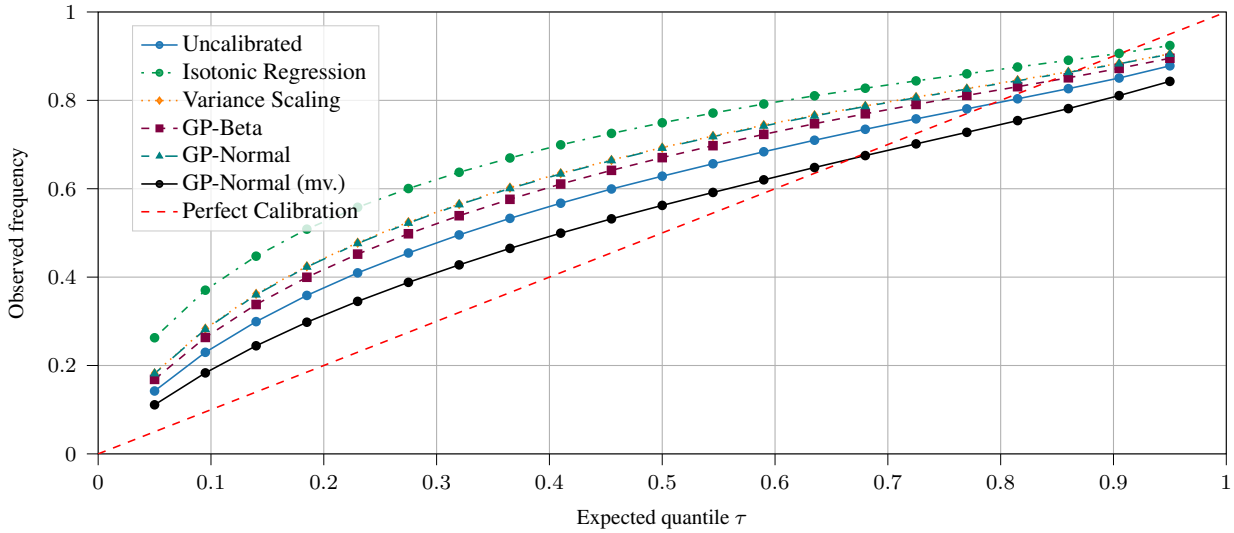
(ii) Reliability diagram for the track c_y coordinate.



(iii) Reliability diagram for the track width.

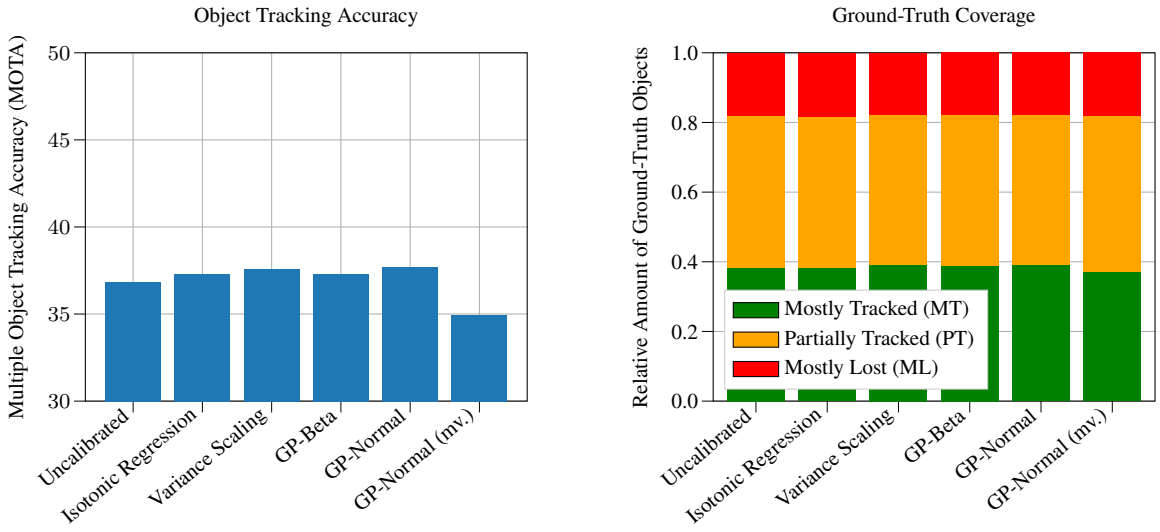


(iv) Reliability diagram for the track height.



(v) Reliability diagram for the joint multivariate quantile.

Figure 6.6: Reliability diagrams (object tracking) of the spatial uncertainty for a `Faster R-CNN` before and after uncertainty calibration. Especially in the reliability diagrams for the single bounding box quantities, the multivariate GP-Normal leads to the best calibration properties. However, this seems to be detrimental for the multivariate quantile where the quantile coverage decreases.



(i) Tracking accuracy before and after calibration.

(ii) Ground-truth coverage before and after calibration.

Figure 6.7: Visualization of the MOT metrics before and after spatial uncertainty calibration. Except for the Isotonic Regression [34], each calibration method is able to keep the trajectory coverage during object tracking. The gain in the MOT tracking accuracy MOTA is not as large as for semantic confidence calibration in Fig. 6.5. In this case, the standard GP-Normal is able to achieve the overall best tracking accuracy.

a dynamic amount of bins that are used for the recalibration of the Cumulative Density Function (CDF). In contrast, the non-parametric GP-Beta [41] is also able to offer better tracking accuracy (MOTA), less ID switches, and an improved NLL score. However, although the GP-Beta is able to calibrate by means of distribution calibration, the method is still limited to a certain type of recalibration functions due to its parametrization. At this point, it is remarkable that the simple Variance Scaling [42, 18], which is nothing else but a Temperature Scaling [13] for the variance of a normal distribution, is also able to achieve good tracking results although it only uses a single parameter for recalibration. In the experiments, our GP-Normal (cf. Sec. 5.4) is able to even further improve the performance of the object tracking. As described in Sec. 5.4, the GP-Normal is a combination of the Variance Scaling and the flexible parameter estimation of the GP-Beta. As already suggested for the spatial uncertainty calibration methods in Sec. 5.5, we can confirm in our evaluations for object tracking that a direct Gaussian fit during uncertainty calibration is advantageous if subsequent applications (such as Kalman filtering in this case) also require normal distributions. In our experiments, we observe a superior recalibration performance of our GP-Normal as it offers a higher flexibility in calibration compared to the standard Variance Scaling. The multivariate GP-Normal with covariance estimation offers the best scores for the spatial quantile calibration (M-QCE), tracking precision (MOTP), and does not lead to a deterioration of the semantic confidence calibration metrics as opposed to the remaining spatial calibration methods. Furthermore, the reliability diagrams in Fig. 6.6 show a good calibration performance of the multivariate GP-Normal. However, it does not lead to the desired improvement in the tracking performance as initially assumed. As already mentioned, the loss in tracking accuracy is mainly caused by the higher amount of ID switches during object tracking. In our experiments, we observe that

the multivariate GP-Normal leads to better calibrated but also narrower prediction intervals. We assume that less observations are assigned to existing tracks, leading to intermediate losses of interrelated object IDs. In contrast, the non-parametric Isotonic Regression leads to wider prediction intervals which might be advantageous for the track assignment on the one hand. On the other hand, this also leads to less tracks that are mostly tracked (cf. Fig. 6.7) as well as to a lower tracking precision MOTP which might be a hint that the recalibrated uncertainty is also not optimal. Since the track association does also depend on the (static) system uncertainty, we assume that an adjustment of the system noise should be done in conjunction with the new spatial calibration methods. This is subject of future work.

Therefore, we conclude that a calibration of the semantic confidence and spatial uncertainty leads to significant improvements in object tracking. We can confirm our initial suggestions from Sec. 5.5 that a direct Gaussian fit during spatial recalibration is advantageous for subsequent applications such as Kalman filtering that require normal distributions as an interface. Furthermore, our position-dependent semantic confidence calibration methods (cf. Sec. 3.3) and especially the multivariate Histogram Binning showed superior performance in our evaluations compared to the standard calibration methods.

6.7 Conclusion for Calibration in Object Tracking

The goal of this chapter is to show a possible use-case of uncertainty calibration for subsequent applications after object detection. For example, in the context of autonomous driving, it is important to not only detect objects but also to track their position over time. This is a crucial part of image-based environment perception. In this chapter, we therefore applied the methods for semantic confidence calibration from Chap. 3 and for spatial uncertainty calibration from Chap. 5 to the task of object tracking. For this reason, we have introduced the concept of tracking-by-detection where an object detector is used to obtain observations of possible objects within a single frame. These observations were used in a recursive Bayesian filtering framework to construct object trajectories that are tracked over multiple subsequent frames. An object tracking model consists of a position tracking as well as a track management with an assessment of the belief that a track is alive, i.e., the belief that it matches a real ground-truth object. Therefore, we have firstly derived a belief model for the object existence that consumes the (calibrated) confidence information of the object detector to make assumptions of the probability for the track existence. Second, we have introduced the concept of Kalman filtering which is an implementation of the recursive filtering using normal distributions. The Kalman filter is used to track the position information of an object. Furthermore, the Kalman filter is also capable to process the uncertainty which is inherent in the observations obtained by the object detector. Thus, we used a probabilistic object detector (cf. Sec. 2.2.2) to obtain individual uncertainty quantifications for each observation. In this context, we have applied our spatial uncertainty calibration methods from Chap. 5 for uncertainty recalibration. Therefore, we have been able to evaluate the effect of intermediate uncertainty calibration on the task of object tracking.

The evaluations show that both types of uncertainty calibration lead to significant improvements in the basic tracking performance as well as in the calibration properties of the track uncertainty. We can show that

semantic confidence calibration leads to improved tracking performance with some limitations regarding the increase of false negatives. Furthermore, our position-dependent calibration methods and especially the multivariate Histogram Binning (cf. Sec. 3.3.1) showed superior performance. Thus, the calibration of the semantic confidence had a positive influence on the management of individual tracks during object tracking. Similarly, the methods for spatial uncertainty calibration also showed an improvement of the object tracking performance as well as the spatial uncertainty representation of the tracked objects. In this case, the non-parametric Isotonic Regression [34] and especially our parametric GP-Normal (cf. Sec. 5.4) showed superior performance when applied to the task of object tracking. Thus, we can confirm our initial assumptions from Sec. 5.5 that Gaussian calibration methods such as Variance Scaling [42, 18] or our GP-Normal are advantageous if used in conjunction with subsequent applications that use normal distributions as input (such as Kalman filtering in our case).

Therefore, we conclude that uncertainty calibration in general has the potential to influence the object tracking positively Furthermore, our extended position-dependent confidence calibration methods (cf. Sec. 3.3) as well as our extended distribution-aware regression calibration methods (cf. Sec. 5.4) were able to further improve the tracking performance as they are a valuable contribution to better reflect the uncertainty of individual samples. For future applications, it might be interesting to investigate the effect of intermediate Bayesian confidence calibration (cf. Sec. 4.1) to the task of object tracking. In this case, the object confidence is represented by a sample distribution obtained by the Bayesian calibration methods. A possible use case might be to adapt a particle filter framework to directly track the confidence distribution over time. For the track management, a track might then be discarded if the expected confidence falls below a certain threshold or if the variance of the confidence distribution gets too high. We let this open for future research as it might be a valuable contribution to the context of object tracking.

7 Conclusion

In this work, we evaluated the consistency of uncertainty quantification in the context of image-based object detection which is a part of environment perception, e.g., for safety-critical applications such as autonomous driving functions. Especially for safety-critical applications, a reliable uncertainty assessment is of special interest. Modern detection algorithms are based on neural networks that aim to identify multiple objects with their position, size, and their class within a single image. However, it is a known issue that modern neural networks tend to produce either overconfident [13] or underconfident [2] uncertainty estimations, depending on their architecture and use-case. Therefore, we started by introducing the basic concepts of object detection and uncertainty quantification in Chap. 2. Furthermore, we evaluated the semantic confidence (Chap. 3) as well as the spatial uncertainty (Chap. 5) for consistency throughout this work which both represent the uncertainty in the class label and the spatial position uncertainty, respectively. Both types of uncertainty are estimated by probabilistic object detection models [40, 71, 16]. Thus, our target was not to evaluate or improve the baseline detection performance (e.g., precision or recall) but to assess if the predicted uncertainties reliably represent the model's uncertainty which is equivalent to the observed error. Besides measuring for uncertainty consistency, i.e., for calibration, we focused on methods which seek to correct possibly uncalibrated uncertainty estimates without the need of retraining a detection model. Thus, it is possible to learn and apply a post-hoc remapping either of semantic or spatial uncertainty to achieve an improved representation of the observed error. In the last Chap. 6, we evaluated the effect of uncertainty calibration on a subsequent object tracking. In contrast to object detection, a tracking model seeks to track the same object across a sequence of frames over time. The object tracking is based on a mathematical framework in which we can integrate our proposed calibration methods. Therefore, we have been able to evaluate our uncertainty calibration methods for the complete perception pipeline from object detection to object tracking.

7.1 Summary of Semantic Confidence Calibration

In Chap. 3, we started with our evaluations for the calibration of the semantic confidence in the context of object detection. Similar to the task of classification, an object detector provides a confidence score but for each detection individually. This confidence can be interpreted as the model's belief of matching a real ground-truth object. In a first step, we extended the definition of confidence calibration to the task of object detection and further introduced a dependency on the object position to the definition of calibration. Thus, it is possible to measure the influence of the additional position information on the calibration of the confidence score. Besides object detection, we also provided the respective calibration definitions for instance and semantic segmentation. On this basis, we extended common calibration methods such as Histogram Binning [36], Logistic Calibration [35], and Beta Calibration [43] to also include additional position information into the recalibration of detection or segmentation models. We extended these methods in a way so that they are capable of capturing possible correlations between confidence, position information, and miscalibration. In the scope of semantic segmentation, we could not find a major improvement in calibration as the examined

models already provide qualitatively well-calibrated confidence estimates. In contrast, we have been able to significantly improve the calibration as well as the mask quality of instance segmentation models. Furthermore, we found the examined object detection models to be miscalibrated which could be improved using our calibration methods. Although we could only find a minor connection between position and miscalibration, our extended methods offered a qualitatively good calibration performance. In Chap. 6, we further evaluated the effect of (position-dependent) confidence calibration on the task of object tracking. For this reason, we derived a framework to estimate the score for the object existence using the confidence provided by the underlying detection model. When semantic confidence calibration was applied as an intermediate step, we observed that calibration in general and especially our position-dependent calibration methods are able to significantly improve the tracking performance as well as the intrinsic track calibration properties. Therefore, we conclude that semantic confidence calibration in general and especially our position-dependent methods are able to enhance the tracking performance as well as the consistency of the track uncertainty which is a valuable contribution especially for safety-critical applications.

7.2 Summary of Bayesian Confidence Calibration

Basically, the methods for semantic confidence calibration are obtained by standard Maximum Likelihood Estimation (MLE). This approach yields deterministic parameters for the calibration functions. However, especially during position-dependent calibration, it might occur that a new sample during inference is out of the known distribution that has been used to learn the calibration parameters within the training phase. In Chap. 4, we therefore proposed the term of Bayesian confidence calibration that introduces additional epistemic (model) uncertainty into a calibration mapping. Similar to Bayesian neural networks, we can construct a calibration function whose parameters are not deterministic but rather represented by probability distributions. During inference, we can sample from these distributions to obtain a sample distribution for each calibrated estimate that represents the epistemic uncertainty of the calibration model. In this context, we utilized Stochastic Variational Inference (SVI) to place variational Gaussian distributions over each calibration parameter and to learn the moments of the variational distribution during calibration training. In this way, we are able to treat the calibration functions in a Bayesian way which allows to construct probability distributions for the calibrated confidences. In our evaluations, we showed that a calibration method learned by SVI is able to offer the same calibration performance compared to the standard methods learned by MLE. Furthermore, the Bayesian calibration models provide an additional uncertainty for the calibrated confidence. We showed that this additional uncertainty might be used as a sufficient criterion to detect a possible covariate shift. Therefore, Bayesian confidence calibration is valuable extension especially for safety-critical applications where a reliable uncertainty assessment of each component is of great importance.

7.3 Summary of Spatial Uncertainty Calibration

After our examinations for the consistency of semantic confidence, in Chap. 5 we focused on the calibration properties of spatial uncertainty. Similar to the semantic confidence, the spatial uncertainty should represent the observed error during inference. In this context, we reviewed several definitions for regression uncertainty calibration and related them to each other. Each of these definitions comes with its own metrics to measure miscalibration that we use in our evaluations later on. Furthermore, we presented common calibration methods in the context of spatial uncertainty calibration that are divided into parametric and non-parametric methods. The parametric calibration methods output a parametric probability distribution (commonly Gaussian) after calibration, whereas the non-parametric methods output a probability distribution of arbitrary shape. Subsequently, we proposed an extended parametric calibration method GP-Normal that is capable to jointly recalibrate multiple dimensions in a single forward pass while capturing possible correlations between these dimensions. In addition, we proposed a covariance estimation scheme which allows to introduce and model correlations between dimensions that have been learned independently from each other. The experiments for the spatial calibration methods showed that the simple non-parametric Isotonic Regression [34] offers the best results when applied to a probabilistic object detector. Additionally, our extended GP-Normal method has also been able to achieve a good calibration performance. When applied to a subsequent object tracking in Chap. 6, we confirmed these observations and further observed that especially the parametric Gaussian calibration methods achieve a good performance. In these experiments, we have been able to confirm our initial assumptions that a direct modeling of a Gaussian distribution is advantageous if a subsequent application such as Kalman filtering also requires a parametric Gaussian as well. Therefore, similar to the task of semantic confidence calibration, we conclude that spatial uncertainty calibration is a good way to improve the uncertainty consistency of a detection as well as a state estimation within object tracking.

7.4 Final Remarks

In this work, we evaluate the uncertainty of detection and tracking models before and after the application of post-hoc calibration methods. In our experiments, we can show that each kind of calibration is a valuable contribution towards consistent uncertainty and thus to an improved assessment of the model's reliability. Our extended calibration metrics and methods from chapters 3 and 5 have shown to be a good contribution for the safety-relevant context. In the final Chap. 6, we have been able to show that the task of calibration is not only relevant for the object detection itself but also advantageous for subsequent applications such as object tracking. Therefore, we conclude that calibration in general and especially our newly proposed methods are precious contributions within a safety-relevant context.

Bibliography

[1] F. Küppers, J. Kronenberger, A. Shantia, and A. Haselhoff, "Multivariate confidence calibration for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 326–327, 2020.

[2] F. Schwaiger, M. Henne, F. Küppers, F. Schmoeller Roza, K. Roscher, and A. Haselhoff, "From black-box to white-box: Examining confidence calibration under different conditions," in *Proceedings of the Workshop on Artificial Intelligence Safety 2021 (SafeAI 2021) co-located with the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021)*, 2021.

[3] F. Küppers, J. Kronenberger, J. Schneider, and A. Haselhoff, "Bayesian confidence calibration for epistemic uncertainty modelling," in *2021 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2021.

[4] F. Küppers, A. Haselhoff, J. Kronenberger, and J. Schneider, *Confidence Calibration for Object Detection and Segmentation*, pp. 225–250. Cham: Springer International Publishing, 2022.

[5] F. Küppers, J. Kronenberger, A. Haselhoff, and J. Schneider, *Fahrerassistenzsysteme und automatisiertes Fahren*, vol. 2394 of *VDI-Berichte*, ch. Calibration of Neural Networks for Detection Models (German original: Kalibrierung von Neuronalen Netzen für Detektionsmodelle), pp. 49–69. VDI Verlag, May 2022.

[6] F. Küppers, J. Schneider, and A. Haselhoff, "Parametric and multivariate uncertainty calibration for regression and object detection," in *European Conference on Computer Vision Workshops*, Springer, Oct. 2022. in press.

[7] F. Sultana, A. Sufian, and P. Dutta, "Advancements in image classification using convolutional neural network," in *2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, pp. 122–129, IEEE, 2018.

[8] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[9] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal Loss for Dense Object Detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988, 2017.

[10] A. Haselhoff, J. Kronenberger, F. Kuppers, and J. Schneider, "Towards black-box explainability with gaussian discriminant knowledge distillation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21–28, 2021.

[11] A. Niculescu-Mizil and R. Caruana, "Predicting good probabilities with supervised learning," in *Proceedings of the 22nd International Conference on Machine Learning*, pp. 625–632, 2005.

[12] M. Naeini, G. Cooper, and M. Hauskrecht, "Obtaining Well Calibrated Probabilities Using Bayesian Binning," in *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, pp. 2901–2907, 2015.

[13] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," in

Proceedings of the 34th International Conference on Machine Learning, vol. 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330, PMLR, August 2017.

[14] Z. Yang, J. Li, and H. Li, "Real-time pedestrian and vehicle detection for autonomous driving," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 179–184, IEEE, 2018.

[15] D. Feng, L. Rosenbaum, C. Glaeser, F. Timm, and K. Dietmayer, "Can we trust you? on calibration of a probabilistic object detector for autonomous driving," in *IEEE/RSJ International Conference on Intelligent Robots and Systems Workshops (IROSW)*, 2019.

[16] D. Feng, A. Harakeh, S. L. Waslander, and K. Dietmayer, "A review and comparative study on probabilistic object detection in autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[17] X. Jiang, M. Osl, J. Kim, and L. Ohno-Machado, "Calibrating predictive model estimates to support personalized medicine," *Journal of the American Medical Informatics Association*, vol. 19, no. 2, pp. 263–274, 2011.

[18] M.-H. Laves, S. Ihler, J. F. Fast, L. A. Kahrs, and T. Ortmaier, "Well-calibrated regression uncertainty in medical imaging with deep learning," in *Medical Imaging with Deep Learning*, pp. 393–412, PMLR, 2020.

[19] A. Mehrtash, W. M. Wells, C. M. Tempany, P. Abolmaesumi, and T. Kapur, "Confidence calibration and predictive uncertainty estimation for deep medical image segmentation," *IEEE Transactions on Medical Imaging*, pp. 1–1, 2020.

[20] K. Banerjee, D. Notz, J. Windelen, S. Gavarraju, and M. He, "Online camera lidar fusion and object detection on hybrid data for autonomous driving," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1632–1638, IEEE, 2018.

[21] J. Dickmann, J. Klappstein, M. Hahn, N. Appenrodt, H.-L. Bloecher, K. Werber, and A. Sailer, "Automotive radar the key technology for autonomous driving: From detection and ranging to environmental understanding," in *2016 IEEE Radar Conference (RadarConf)*, pp. 1–6, IEEE, 2016.

[22] Y. Li and J. Ibanez-Guzman, "Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems," *IEEE Signal Processing Magazine*, vol. 37, no. 4, pp. 50–61, 2020.

[23] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE transactions on pattern analysis and machine intelligence*, vol. 34, no. 4, pp. 743–761, 2011.

[24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox detector," in *European Conference on Computer Vision (ECCV)*, pp. 21–37, Springer, 2016.

[25] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*,

pp. 779–788, 2016.

[26] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[27] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[29] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587, 2014.

[30] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.

[31] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[32] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[33] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 801–818, 2018.

[34] V. Kuleshov, N. Fenner, and S. Ermon, "Accurate Uncertainties for Deep Learning Using Calibrated Regression," in *International Conference on Machine Learning (ICML)*, pp. 2801–2809, 2018.

[35] J. Platt, "Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods," *Advances in Large Margin Classifiers*, pp. 61–74, 1999.

[36] B. Zadrozny and C. Elkan, "Obtaining Calibrated Probability Estimates from Decision Trees and Naive Bayesian Classifiers," in *Proceedings of the Eighteenth International Conference on Machine Learning (ICML)*, pp. 609–616, 2001.

[37] K. Maag, M. Rottmann, and H. Gottschalk, "Time-dynamic estimates of the reliability of deep semantic segmentation networks," in *2020 IEEE 32nd International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 502–509, IEEE, 2020.

[38] M. Schubert, K. Kahl, and M. Rottmann, "Metadetect: Uncertainty quantification and prediction quality estimates for object detection," in *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–10, IEEE, 2021.

[39] A. Kendall and Y. Gal, "What Uncertainties Do We Need in Bayesian Deep Learning for Computer

Vision?," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 5574–5584, 2017.

[40] Y. He, C. Zhu, J. Wang, M. Savvides, and X. Zhang, "Bounding box regression with uncertainty for accurate object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2888–2897, 2019.

[41] H. Song, T. Diethe, M. Kull, and P. Flach, "Distribution calibration for regression," in *Proceedings of the 36th International Conference on Machine Learning*, vol. 97 of *Proceedings of Machine Learning Research*, (Long Beach, California, USA), pp. 5897–5906, PMLR, 09–15 Jun 2019.

[42] D. Levi, L. Gispan, N. Giladi, and E. Fetaya, "Evaluating and calibrating uncertainty prediction in regression tasks," *arXiv preprint*, vol. abs/1905.11659, 2019.

[43] M. Kull, T. Silva Filho, and P. Flach, "Beta calibration: a well-founded and easily implemented improvement on logistic calibration for binary classifiers," in *Artificial Intelligence and Statistics*, pp. 623–631, 2017.

[44] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision (ECCV)*, pp. 740–755, Springer, 2014.

[45] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[46] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[47] F. Van Der Heijden, R. P. Duin, D. De Ridder, and D. M. Tax, *Classification, parameter estimation and state estimation: an engineering approach using MATLAB*. John Wiley & Sons, 2005.

[48] K. Maag, *Prediction Rating and Performance Improvement for Segmentation Networks by Time-Dynamic Uncertainty Estimates*. PhD thesis, Universität Wuppertal, Fakultät für Mathematik und Naturwissenschaften ..., 2021.

[49] P. Colling, *Uncertainty Quantification and its Applications for Multimodal Semantic Segmentation*. PhD thesis, Universität Wuppertal, Fakultät für Mathematik und Naturwissenschaften ..., 2022.

[50] V. Vapnik, "Principles of risk minimization for learning theory," *Advances in neural information processing systems*, vol. 4, 1991.

[51] J. Aldrich, "Ra fisher and the making of maximum likelihood 1912-1922," *Statistical science*, vol. 12, no. 3, pp. 162–176, 1997.

[52] S. Kullback and R. A. Leibler, "On information and sufficiency," *The annals of mathematical statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[53] C. E. Shannon, "A mathematical theory of communication," *The Bell system technical journal*, vol. 27,

no. 3, pp. 379–423, 1948.

[54] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, pp. 177–186, Springer, 2010.

[55] R. Ge, F. Huang, C. Jin, and Y. Yuan, "Escaping from saddle points—online stochastic gradient for tensor decomposition," in *Conference on learning theory*, pp. 797–842, PMLR, 2015.

[56] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[57] V. N. Vapnik, "Statistical learning theory. adaptive and learning systems for signal processing," *Communications and Control*, vol. 2, pp. 1–740, 1998.

[58] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning," in *International Conference on Machine Learning (ICML)*, pp. 1050–1059, 2016.

[59] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, pp. 448–456, PMLR, 2015.

[60] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in *Competition and cooperation in neural nets*, pp. 267–285, Springer, 1982.

[61] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 91–99, 2015.

[62] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017.

[63] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods," *Machine Learning*, vol. 110, no. 3, pp. 457–506, 2021.

[64] A. Der Kiureghian and O. Ditlevsen, "Aleatory or epistemic? does it matter?," *Structural safety*, vol. 31, no. 2, pp. 105–112, 2009.

[65] D. J. MacKay, "A practical bayesian framework for backpropagation networks," *Neural computation*, vol. 4, no. 3, pp. 448–472, 1992.

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

[67] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in neural information processing systems*, pp. 6402–6413, 2017.

[68] M. D. Hoffman, D. M. Blei, C. Wang, and J. Paisley, "Stochastic variational inference," *Journal of Machine Learning Research*, vol. 14, no. 4, pp. 1303–1347, 2013.

[69] J. Postels, F. Ferroni, H. Coskun, N. Navab, and F. Tombari, "Sampling-free epistemic uncertainty estimation using approximated variance propagation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2931–2940, 2019.

[70] A. Harakeh, M. Smart, and S. L. Waslander, "Bayesod: A bayesian approach for uncertainty estimation in deep object detectors," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 87–93, IEEE, 2020.

[71] D. Hall, F. Dayoub, J. Skinner, H. Zhang, D. Miller, P. Corke, G. Carneiro, A. Angelova, and N. Sünderhauf, "Probabilistic object detection: Definition and evaluation," in *The IEEE Winter Conference on Applications of Computer Vision*, pp. 1031–1040, 2020.

[72] T. Gneiting, F. Balabdaoui, and A. E. Raftery, "Probabilistic forecasts, calibration and sharpness," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 69, no. 2, pp. 243–268, 2007.

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

[74] M. Kull and P. Flach, "Novel decompositions of proper scoring rules for classification: Score adjustment as precursor to calibration," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 68–85, Springer, 2015.

[75] M. H. DeGroot and S. E. Fienberg, "The comparison and evaluation of forecasters," *Journal of the Royal Statistical Society. Series D (The Statistician)*, vol. 32, no. 1/2, pp. 12–22, 1983.

[76] A. Kumar, S. Sarawagi, and U. Jain, "Trainable calibration measures for neural networks from kernel mean embeddings," in *International Conference on Machine Learning*, pp. 2805–2814, PMLR, 2018.

[77] J. Mukhoti, V. Kulharia, A. Sanyal, S. Golodetz, P. H. Torr, and P. K. Dokania, "Calibrating deep neural networks using focal loss," in *Advances in Neural Information Processing Systems*, 2020.

[78] M. Minderer, J. Djolonga, R. Romijnders, F. Hubis, X. Zhai, N. Houlsby, D. Tran, and M. Lucic, "Revisiting the calibration of modern neural networks," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[79] I. O. Tolstikhin, N. Houlsby, A. Kolesnikov, L. Beyer, X. Zhai, T. Unterthiner, J. Yung, A. Steiner, D. Keysers, J. Uszkoreit, *et al.*, "Mlp-mixer: An all-mlp architecture for vision," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[80] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[81] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255, Ieee, 2009.

[82] A. Malinin and M. Gales, "Predictive uncertainty estimation via prior networks," *Advances in neural information processing systems*, vol. 31, 2018.

[83] J. Bröcker, "Reliability, sufficiency, and the decomposition of proper scores," *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography*, vol. 135, no. 643, pp. 1512–1519, 2009.

[84] Y. Ovadia, E. Fertig, J. Ren, Z. Nado, D. Sculley, S. Nowozin, J. Dillon, B. Lakshminarayanan, and J. Snoek, "Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift," *Advances in neural information processing systems*, vol. 32, 2019.

[85] B. Zadrozny and C. Elkan, "Transforming Classifier Scores into Accurate Multiclass Probability Estimates," in *Proceedings of the Eighth International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada*, pp. 694–699, 2002.

[86] M. Naeini and G. Cooper, "Binary Classifier Calibration Using an Ensemble of Near Isotonic Regression Models," in *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 360–369, December 2016.

[87] M. Kull, M. Perello Nieto, M. Kängsepp, T. Silva Filho, H. Song, and P. Flach, "Beyond temperature scaling: Obtaining well-calibrated multi-class probabilities with dirichlet calibration," *Advances in neural information processing systems*, vol. 32, 2019.

[88] A. Kumar, P. S. Liang, and T. Ma, "Verified uncertainty calibration," in *Advances in Neural Information Processing Systems 32*, pp. 3792–3803, Curran Associates, Inc., 2019.

[89] B. Ji, H. Jung, J. Yoon, K. Kim, and y. Shin, "Bin-wise temperature scaling (bts): Improvement in confidence calibration performance through simple scaling techniques," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 4190–4196, 2019.

[90] J. Wenger, H. Kjellström, and R. Triebel, "Non-parametric calibration for classification," in *International Conference on Artificial Intelligence and Statistics*, pp. 178–190, 2020.

[91] K. Gupta, A. Rahimi, T. Ajanthan, T. Mensink, C. Sminchisescu, and R. Hartley, "Calibration of neural networks using splines," in *International Conference on Learning Representations*, 2021.

[92] G. Pereyra, G. Tucker, J. Chorowski, Lukasz Kaiser, and G. Hinton, "Regularizing neural networks by penalizing confident output distributions," *CoRR*, 2017.

[93] M. Rottmann, P. Colling, T. P. Hack, R. Chan, F. Hüger, P. Schlicht, and H. Gottschalk, "Prediction error meta classification in semantic segmentation: Detection via aggregated dispersion measures of softmax probabilities," in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–9, IEEE, 2020.

[94] D. Barber, *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[95] D. L. Libby and M. R. Novick, "Multivariate generalized beta distributions with applications to utility

assessment," *Journal of Educational Statistics*, vol. 7, no. 4, pp. 271–294, 1982.

[96] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2." `https://github.com/facebookresearch/detectron2`, 2019.

[97] A. Kirillov, Y. Wu, K. He, and R. Girshick, "Pointrend: Image segmentation as rendering," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9799–9808, 2020.

[98] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5693–5703, 2019.

[99] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, *et al.*, "Deep high-resolution representation learning for visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 10, pp. 3349–3364, 2020.

[100] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," in *European conference on computer vision*, pp. 173–190, Springer, 2020.

[101] Y. Gal, "Uncertainty in deep learning," *University of Cambridge*, vol. 1, p. 3, 2016.

[102] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine learning*, vol. 37, no. 2, pp. 183–233, 1999.

[103] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[104] T. Pearce, A. Brintrup, M. Zaki, and A. Neely, "High-quality prediction intervals for deep learning: A distribution-free, ensembled approach," in *International Conference on Machine Learning*, pp. 4075–4084, 2018.

[105] M. Fasiolo, S. N. Wood, M. Zaffran, R. Nedellec, and Y. Goude, "Fast calibrated additive quantile regression," *Journal of the American Statistical Association*, vol. 116, no. 535, pp. 1402–1412, 2021.

[106] Y. Chung, W. Neiswanger, I. Char, and J. Schneider, "Beyond pinball loss: Quantile methods for calibrated uncertainty quantification," in *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[107] P. Cui, W. Hu, and J. Zhu, "Calibrated reliable regression using maximum mean discrepancy," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[108] D. Bhatt, K. Mani, D. Bansal, K. Murthy, H. Lee, and L. Paull, "f-cal: Calibrated aleatoric uncertainty estimation from neural networks for robot perception," *arXiv preprint arXiv:2109.13913*, 2021.

[109] I. Steinwart and A. Christmann, "Estimating conditional quantiles with the help of the pinball loss," *Bernoulli*, vol. 17, no. 1, pp. 211–225, 2011.

[110] P. C. Mahalanobis, "On the generalized distance in statistics," *Proceedings of the National Institute*

of Sciences (Calcutta), vol. 2, pp. 49–55, 1936.

[111] A. SenGupta, "Tests for standardized generalized variances of multivariate normal populations of possibly different dimensions," *Journal of Multivariate Analysis*, vol. 23, no. 2, pp. 209–219, 1987.

[112] A. SenGupta, "Generalized variance," *Encyclopedia of statistical sciences*, vol. 6053, 2004.

[113] V. Glivenko, "Sulla determinazione empirica delle leggi di probabilita," *Gion. Ist. Ital. Attauri.*, vol. 4, 1933.

[114] H. G. Tucker, "A generalization of the glivenko-cantelli theorem," *The Annals of Mathematical Statistics*, vol. 30, no. 3, pp. 828–830, 1959.

[115] M. Alvarez and N. Lawrence, "Sparse convolved gaussian processes for multi-output regression," *Advances in neural information processing systems*, vol. 21, 2008.

[116] P. Moreno-Muñoz, A. Artés, and M. Alvarez, "Heterogeneous multi-output gaussian process prediction," *Advances in neural information processing systems*, vol. 31, 2018.

[117] G. Skolidis and G. Sanguinetti, "Bayesian multitask classification with gaussian process priors," *IEEE Transactions on Neural Networks*, vol. 22, no. 12, 2011.

[118] L. Song, X. Zhang, A. Smola, A. Gretton, and B. Schölkopf, "Tailoring density estimation via reproducing kernel moment matching," in *Proceedings of the 25th international conference on Machine learning*, pp. 992–999, 2008.

[119] P. Goovaerts *et al.*, *Geostatistics for natural resources evaluation*. Oxford University Press on Demand, 1997.

[120] E. Snelson and Z. Ghahramani, "Sparse gaussian processes using pseudo-inputs," *Advances in neural information processing systems*, vol. 18, p. 1257, 2006.

[121] J. Hensman, A. Matthews, and Z. Ghahramani, "Scalable variational gaussian process classification," in *Artificial Intelligence and Statistics*, pp. 351–360, PMLR, 2015.

[122] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell, "Bdd100k: A diverse driving video database with scalable annotation tooling," *CoRR*, 2018.

[123] M. Andriluka, S. Roth, and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2008.

[124] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool, "Robust tracking-by-detection using a detector confidence particle filter," in *2009 IEEE 12th International Conference on Computer Vision*, pp. 1515–1522, IEEE, 2009.

[125] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and data fusion*, vol. 11. YBS publishing Storrs, CT, USA:, 2011.

[126] J. Huang and W. Zhou, "Re 2 ema: regularized and reinitialized exponential moving average for target

model update in object tracking," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8457–8464, 2019.

[127] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[128] K. Bernardin and R. Stiefelhagen, "Evaluating multiple object tracking performance: the clear mot metrics," *EURASIP Journal on Image and Video Processing*, vol. 2008, pp. 1–10, 2008.

[129] E. Ristani, F. Solera, R. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *European conference on computer vision*, pp. 17–35, Springer, 2016.