Bergische Universität Wuppertal

Fakultät für Mathematik und Naturwissenschaften

Institute of Mathematical Modelling, Analysis and Computational
Mathematics (IMACM)

Preprint BUW-IMACM 20/49

Marius Schubert, Karsten Kahl and Matthias Rottmann

MetaDetect: Uncertainty Quantification and Prediction Quality Estimates for Object Detection

October 28, 2020

http://www.imacm.uni-wuppertal.de

MetaDetect: Uncertainty Quantification and Prediction Quality Estimates for Object Detection

Marius Schubert[1], Karsten Kahl[1] and Matthias Rottmann[1]

[1]University of Wuppertal, School of Mathematics and Natural Sciences
{schubert,kkahl,rottmann}@math.uni-wuppertal.de

Abstract

In object detection with deep neural networks, the box-wise objectness score tends to be overconfident, sometimes even indicating high confidence in presence of inaccurate predictions. Hence, the reliability of the prediction and therefore reliable uncertainties are of highest interest. In this work, we present a post processing method that for any given neural network provides predictive uncertainty estimates and quality estimates. These estimates are learned by a post processing model that receives as input a hand-crafted set of transparent metrics in form of a structured dataset. Therefrom, we learn two tasks for predicted bounding boxes. We discriminate between true positives ($IoU \geq 0.5$) and false positives ($IoU < 0.5$) which we term meta classification, and we predict IoU values directly which we term meta regression. The probabilities of the meta classification model aim at learning the probabilities of success and failure and therefore provide a modelled predictive uncertainty estimate. On the other hand, meta regression gives rise to a quality estimate. In numerical experiments, we use the publicly available YOLOv3 network and the Faster-RCNN network and evaluate meta classification and regression performance on the Kitti, Pascal VOC and COCO datasets. We demonstrate that our metrics are indeed well correlated with the IoU. For meta classification we obtain classification accuracies of up to 98.92% and AUROCs of up to 99.93%. For meta regression we obtain an R^2 value of up to 91.78%. These results yield significant improvements compared to other network's objectness score and other baseline approaches. Therefore, we obtain more reliable uncertainty and quality estimates which is particularly interesting in the absence of ground truth.

1. Introduction

In recent years, deep neural networks have surpassed other approaches in many tasks with unstructured data. However their in-transparent nature poses many questions and problems. In particular in safety-critical applications, the reliability of deep neural network predictions is of highest interest as neural networks tend to provide high confidences even when they fail [13]. In order to detect these and therefore make neural networks more interpretable, meaningful uncertainties are required [26]. A broad survey on uncertainty in machine learning can be found in [19]. A very important and also reducible type of uncertainty is the model uncertainty resulting from the fact that the ideal parameters are unknown and have to be estimated from data [26, 2, 11, 6]. Bayesian models consider these uncertainties [26]. However, as Bayesian models for deep learning are nowadays infeasible, many different frameworks based on variational approximations have been proposed [2, 6, 11]. For instance, Monte-Carlo (MC) Dropout [11] is used as a feasible approximation to Bayesian inference.

In classification tasks, a natural approach to detect incorrect predictions / false positives is to introduce a dispersion measure on the softmax probabilities, such as the entropy or one minus the highest softmax probability and then threshold on this [23, 17]. As in the works [35, 34, 4, 36, 25] that extend [17] to semantic segmentation, we refer to this task as *meta classification*. A good meta classification performance requires well adjusted predictive uncertainty measures. Variational approximations of Bayesian learning are one approach to this. However, also feasible ones like [11] still require numerous inferences of one and the same input in order to estimate predictive uncertainty. In object detection, the network's objectness score is subject to thresholding which in a natural sense can be considered built in meta classification as part of state-of-the-art object detection pipelines [32, 33].

Confidence calibration has been proposed to account for

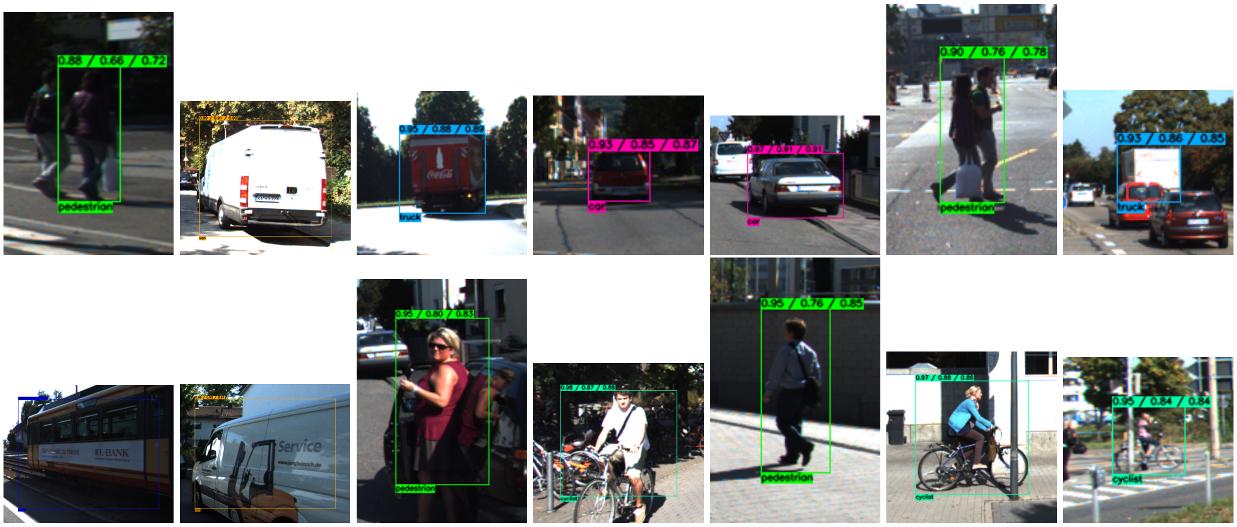


Figure 1: Examples of predicted bounding boxes with objectness score / true IoU / predicted IoU. The predictions, and therefore the data to train and evaluate gradient boosting, are made with the YOLOv3 network, the Kitti dataset and a score threshold $t = 0.01$.

the miscalibration of the objectness score [30, 14]. However, most approaches (e.g. histogram-based ones [21]) calibrate the score in a statistical sense. This is not sufficient to account for the correctness or quality of a single predicted bounding box. Therefore, uncertainty estimates for objects detection networks seem more promising.

In [27] a baseline for meta classification in object detection is presented. The localization variables are determined using the candidate boxes that are present after the score thresholding and before the non-maximum suppression (NMS). The mean values of the associated candidate boxes represent the predicted bounding box and the added variances represent the localization uncertainty. In combination with the class uncertainty, which is generated from the entropy of the class probabilities in a softmax output, an uncertainty measure is obtained for each box. A threshold is used to decide whether the box is accepted or rejected as a prediction. To generate the uncertainty not only from the output, in [20, 22, 16, 15] the loss is changed in such a way that the uncertainties of the localization variables are learned and displayed in their own newly introduced neurons. In [20, 15] MC-Dropout is used for the localization and variance calculation of the predicted bounding boxes. The uncertainty introduced in the loss combined with the localization variance is applied to the individual anchor boxes/priors. In [15] the NMS is replaced by a Bayesian inference, in particular the assignment is done with a different cluster method. The performance with respect to prediction accuracy for all the listed performance metrics (mAP, PDQ Score, mGMUE, mCMUE) increases compared to other state of the art methods introduced in [20, 22, 28, 27, 9, 10]. In [28], classification uncertainty is extracted from object detection networks via dropout sampling.

For semantic segmentation, a number of works that estimate the segmentation quality have been introduced [18, 37, 29, 34, 36, 25, 4]. These works make use of the richness of information of the segmentation networks output, as there is a probability distribution available in each pixel of a given segment. In object detection tasks, this richness is not present.

Our contribution. In this work we extend the false positive detection baseline of [17] from classification to object detection. As the works [34, 36, 25, 4] in semantic segmentation we consider two (meta) tasks: Discrimination between true positive ($IoU \geq 0.5$) and false positive ($IoU < 0.5$) boxes which is termed *meta classification* and regression with IoU values which is termed *meta regression*. To approach these tasks without significantly increasing the computational cost caused object detection inferences, we introduce a post processing framework that in general can be added to any object detection network. Only by means of the network's output, our framework trains two models, i.e., one for each meta task. More precisely, we construct a set of handcrafted and interpretable metrics that may reveal the network's uncertainty about a prediction. Amongst others, this includes the number of candidate boxes before NMS that overlap with the given predicted box to a chosen extent, the score, the class probability distribution, the boxes size and aspect ratio and many others. If desired, these metrics can be extended by MC-dropout statistics. In general any object-wise metric that seems to be helpful can be added. From this we obtain a structured dataset where the rows amount to the predicted objects in a given number of images and the columns amount to the constructed metrics. From this set of metrics we learn both meta tasks with different classical machine learning models, i.e., linear ones, shallow neural networks and gradient boosting. Figure 1 illustrates the effect of meta regression. In the depicted cases, the bounding boxes are predicted with an overconfident score, whereas the true IoU is significantly lower and the predicted IoU is close to the true one.

In our numerical experiments, we study the correlation of our constructed metrics with the IoU of prediction and ground truth, we study different sets of metrics and compare those to a score baseline. For both meta tasks, we significantly outperform the baseline. This is observed consistently over different datasets (Kitti [12], Pascal VOC [7] and COCO [24]) and different object detection networks (YOLOv3 [32] and Faster-RCNN [33]). We do not observe an improvement when performing Monte-Carlo dropout with a modified YOLOv3 network and including the obtained uncertainties into our set of metrics. This strengthens the statement, that our approach is reliable.

Related work. In this section, we clarify the differences between related works and ours. The idea behind this work is in spirit similar to the approaches for semantic segmentation [18, 37, 29, 34, 36, 25, 4], however the nature of the output provided by segmentation networks and object detection networks is so different, such that the resulting uncertainty metrics are also clearly different.

Our approach is solely based on post processing. One can plug-in MC dropout quantities if the network architecture allows, however, this is not mandatory. On the other hand, [20, 22, 16, 15] incorporate the uncertainty quantification into the original network and change their loss functions and also the ultimate layer. This requires additional training and does not aim at quantifying the uncertainty of the original network. In addition, as opposed to the other works we provide an simple and modular statistical benchmark suite that can be used for any object detection network in combination with any object-wise uncertainty quantification method to obtain performance indicators for the latter.

In [27] box-wise uncertainty information is calculated by means of associated candidate boxes. Whether a prediction is accepted or rejected is decided by simply thresholding on the resulting measure. Beyond that, neither meta regression, meta classification nor a general statistical evaluation of uncertainty measures is performed. Instead they use their uncertainty measure to improve the non maximum suppression. In [30] introduces a confidence calibration method that uses variances from a Bayesian neural network trained on the confidence calibration task. This aims at estimating the uncertainty, given the localization variables and a specific confidence level, whereas we estimate predictive uncertainty for a given input. Meta classification was not performed. In [28] the authors extract classification, localization and spatial uncertainty under dropout sampling and turn this into an overall performance improvement. Also in that work, there is no meta classification or regression performed and the proposed concept requires dropout inference.

Outline. The remainder of this work is structured as follows: In section 2 we introduce the concept of object detection and explain how to perform uncertainty quantification by means of candidate boxes and MC dropout. This is followed by the construction of uncertainty metrics using predictive uncertainty and geometry information in section 3. Afterwards we present numerical results in section 4, benchmarking several approaches against each other. First, we present the chosen parameters and datasets we have used for our experiments with the YOLOv3 and Faster-RCNN networks. Afterwards we discuss correlation coefficients and evaluate the usefulness of our box-wise uncertainty metrics for meta classification and regression.

2. The Generic Object Detection Pipeline

In this section we briefly review the concept of object detection and its commonly used components, as they will be of interest for our uncertainty quantification method and experiments with those.

Any image x from the training or validation data is equipped with a set y containing G ground truth bounding boxes, i.e.,

$$y = \{b^{(i)} : i = 1, ..., G\}\,, \tag{1}$$

where each ground truth box is a tuple

$$b^{(i)} = (r^{(i)}_{\min}, r^{(i)}_{\max}, c^{(i)}_{\min}, c^{(i)}_{\max}, \kappa^{(i)}) \tag{2}$$

containing minimal and maximal row and column indices $r^{(i)}_{\min}, r^{(i)}_{\max}, c^{(i)}_{\min}$ and $c^{(i)}_{\max}$ as well as a class index $\kappa^{(i)} \in \{1, \ldots, C\}$. Given an image x, a neural network f computes a set of a fixed number N of inferred bounding boxes (or candidate boxes)

$$f(x) = \hat{y}_c = \{\hat{b}^{(i)} : i = 1, \ldots, N\} \tag{3}$$

where each $\hat{b}^{(i)}$ is a tuple of estimated values

$$\hat{b}^{(i)} = (\hat{r}^{(i)}_{\min}, \hat{r}^{(i)}_{\max}, \hat{c}^{(i)}_{\min}, \hat{c}^{(i)}_{\max}, s^{(i)}, p^{(i)}_1, \ldots, p^{(i)}_C)\,. \tag{4}$$

Therein, the initial four values have the same meaning as for the ground truth, $s^{(i)}$ denotes the so-called (objectness) score which takes values in $[0, 1]$ and (roughly) indicates how likely it is that $b^{(i)}$ contains an object. Furthermore, given an input x the network f provides class probabilities $p_1^{(i)}, \dots, p_C^{(i)}$. The predicted class is given by $\hat{\kappa}^{(i)} = \arg\max_{k=1,\dots,C} p_k^{(i)}$.

A typical performance measure, that indicates to which extent a prediction is in accordance to the ground truth, is the IoU. By defining the sets of pixels contained in a ground truth box $b^{(i)}$, that is $\beta^{(i)} = \{(r,c) : r_{\min}^{(i)} \leq r \leq r_{\max}^{(i)},\ c_{\min}^{(i)} \leq c \leq c_{\max}^{(i)}\}$, and analogously defining $\hat{\beta}^{(j)}$ for a estimated box $\hat{b}^{(j)}$, the IoU of $\hat{b}^{(i)}$ and $b^{(j)}$ is defined by

$$IoU(\hat{b}^{(i)}, b^{(j)}) = \frac{|\hat{\beta}^{(i)} \cap \beta^{(j)}|}{|\hat{\beta}^{(i)} \cup \beta^{(j)}|}\,, \tag{5}$$

where $\hat{\kappa}^{(i)} = \kappa^{(j)}$.

Downstream of a typical object detection pipeline, the *candidate boxes* $\hat{y}_c$ are filtered by discarding all boxes $\hat{b}^{(i)}$ whose corresponding estimated score values $s^{(i)}$ remain below a chosen threshold t. That is, we define *filtered candidate boxes*

$$\hat{y}_s = \{\hat{b}^{(i)} \in \hat{y}_c : \hat{s}^{(i)} \geq t\}\,. \tag{6}$$

Typically this is followed by the so-called *non-maximum suppression* (NMS).

Let initially $\hat{y} = \emptyset$. Iteratively, the estimated box $\hat{b}^{(i)} \in \hat{y}_s$ with maximal score $s^{(i)}$ is determined and all other boxes $\hat{b}^{(j)}$ with $\hat{\kappa}^{(i)} = \hat{\kappa}^{(j)}$ and $IoU(\hat{b}^{(i)}, \hat{b}^{(j)}) \geq \tau$, where τ is a specific threshold, are discarded from $\hat{y}_s$. The box $\hat{b}^{(i)}$ is also removed from $\hat{y}_s$ and added to $\hat{y}$. This is repeated until $\hat{y}_s = \emptyset$. Afterwards, $\hat{y}$ represents the set of predicted boxes. For a more precise description of specific object detection pipelines, we refer to [31, 32, 33].

3. Uncertainty Metrics for Object Detection

In this section we construct uncertainty metrics for every $\hat{b}^{(i)} \in \hat{y}$. We do so in two stages, first by introducing the general metrics that can be obtained from the object detection pipeline as is. Second, we extend this by additional metrics that can be computed when using MC dropout.

We consider a predicted bounding box $\hat{b}^{(i)} \in \hat{y}$ and its corresponding filtered candidate boxes $\hat{b}^{(j)} \in \hat{y}_s$, that were discarded by the NMS. + The number of corresponding candidate boxes $\hat{b}^{(j)} \in \hat{y}_s$ filtered by the NMS intuitively gives rise to the likelihood of observing a true positive. The more candidate boxes $\hat{b}^{(j)}$ belong to $\hat{b}^{(i)}$, the more likely it seems that $\hat{b}^{(i)}$ is a true positive. We denote by $N^{(i)}$ the number of candidate boxes $\hat{b}^{(j)}$ belonging to $\hat{b}^{(i)}$ but suppressed by NMS. We increment this number by 1 and also count in $\hat{b}^{(i)}$.

For a given image x we have the set of predicted bounding boxes $\hat{y}$ and the ground truth y. As we want to calculate values that represent the quality of the neural network's prediction, we first have to define uncertainty metrics for the predicted bounding boxes in $\hat{y}$. For each $\hat{b}^{(i)} \in \hat{y}$, we define the following quantities:

- the number of candidate boxes $N^{(i)} \geq 1$ that belong to $\hat{b}^{(i)}$ ($\hat{b}^{(i)}$ belongs to itself),
- the predicted box $\hat{b}^{(i)}$, i.e., the values of the tuple $(\hat{r}_{\min}^{(i)}, \hat{r}_{\max}^{(i)}, \hat{c}_{\min}^{(i)}, \hat{c}_{\max}^{(i)}, s^{(i)}, p_1^{(i)}, \dots, p_C^{(i)}) \in \mathbb{R}^{5+C}$,
- the size $d = (\hat{r}_{\max}^{(i)} - \hat{r}_{\min}^{(i)}) \cdot (\hat{c}_{\max}^{(i)} - \hat{c}_{\min}^{(i)})$ and the circumference $g = 2 \cdot (\hat{r}_{\max}^{(i)} - \hat{r}_{\min}^{(i)}) + 2 \cdot (\hat{c}_{\max}^{(i)} - \hat{c}_{\min}^{(i)})$,
- IoU_{pb}: the IoU of $\hat{b}^{(i)}$ and the box with the second highest score that was suppressed by $\hat{b}^{(i)}$. This values is zero if there are no boxes corresponding to $\hat{b}^{(i)}$ suppressed by the NMS ($N^{(i)} = 1$),
- the minimum, maximum, arithmetic mean and standard deviation for all $(\hat{r}_{\min}^{(i)}, \hat{r}_{\max}^{(i)}, \hat{c}_{\min}^{(i)}, \hat{c}_{\max}^{(i)}, s^{(i)})$, size d and circumference g from $\hat{b}^{(i)}$ and all the filtered candidate boxes that were discarded from $\hat{b}^{(i)}$ in the NMS,
- the minimum, maximum, arithmetic mean and standard deviation for the IoU of $\hat{b}^{(i)}$ and all the candidate boxes corresponding to $\hat{b}^{(i)}$ that suppressed in the NMS,
- relative sizes $rd = d/g$, $rd_{\min} = d/g_{\min}$, $rd_{\max} = d/g_{\max}$, $rd_{mean} = d/g_{mean}$ and $rd_{std} = d/g_{std}$,
- the IoU of $\hat{b}^{(i)}$ and the ground truth $b^{(i)}$, this is not an input to a meta model but serves as the ground truth provided to the respective loss function.

Altogether, this results in $46 + C$ uncertainty metrics.

We now elaborate on how to calculate uncertainty metrics for every $\hat{b}^{(i)} \in \hat{y}$ when using MC dropout. To this end, we consider the bounding box $\hat{b}^{(i)} \in \hat{y}$ that was predicted without dropout and then we observe under dropout J times the output of the same anchor box that produced $\hat{b}^{(i)}$ and denote them by $\hat{b}_1^{(i)}, ..., \hat{b}_J^{(i)}$. For these $J + 1$ boxes we calculate the minimum, the maximum, the arithmetic mean and the standard deviation for the localization variables and the objectness score over $\hat{b}^{(i)}, \hat{b}_1^{(i)}, ..., \hat{b}_J^{(i)}$. This is done for every $\hat{b}^{(i)} \in \hat{y}$ and results in 20 additional dropout uncertainty metrics. This means $66 + C$ uncertainty metrics in total. Executing this procedure for all available test images we end up with a structured dataset. The number of predicted objects constitutes to the number of rows and the columns are given by the registered metrics. After defining a training / test splitting of this dataset, we learn meta classification ($IoU \geq 0.5$ vs. $IoU < 0.5$) and meta regression (quantitative IoU prediction) from the training part of this data.

All these presented metrics, except for the true IoU can be computed without the knowledge of the ground truth. Our aim is to analyze to which extent they are suited for the tasks of meta classification and meta regression.

4. Numerical Experiments: Pascal VOC, KITTI and COCO

In this section we investigate the properties of the metrics defined in the previous sections for the example of object detection for three different datasets and two different networks. We deploy the YOLOv3 network [32] for which we use a reference implementation in Tensorflow [1] as well as weights self-trained on the Kitti dataset [12], the Pascal VOC2007 dataset [7] and the COCO dataset [24]. For Kitti we split the labelled training images as for the test images are no labels available. The images 0-5480 are used to train our network and the last 2000 (image 5481-7480) are used to evaluate our method. For Pascal VOC we used the training images from the years 2007 to 2011 to train our network and we evaluate our method on the 4952 Pascal VOC2007 test images. For COCO we train on all COCO2014 training images and evaluate our method on 2500 randomly selected test images from the COCO2014 test images. For the Faster-RCNN [33] we use a reference implementation in Tensorflow as well as pre-trained weights for all three datasets. When evaluating our method on the Kitti dataset we use all available labelled images. The Faster-RCNN is trained exclusively for the two classes “person” and “car”. Eventually, the Faster-RCNN might have overfitted the training data. Indeed we observe high prediction accuracy. For Pascal VOC and COCO we use the same images as for our tests with the YOLOv3 network.

For more information about the default training and test parameters that we do not deviate from, we refer to the publicly available source codes and to [31, 32, 33] for a detailed explanation of the respective parameters.

We evaluate our methods for meta classification and regression on 33 different score thresholds, starting at 0.01, continuing with thresholds equal to $k/40$ until reaching 0.8 ($k = 1, \dots, 32$). Over the course of thresholds, the highest mean average precision (mAP) values obtained by YOLOv3 are 91.99% on Kitti, 86.9% on Pascal VOC and 61.13% on COCO. For a given class, the average precision (AP) is the area under precision recall curve. Mean average precision is the mean of the AP values for all considered classes. For all three datasets the mAP is equal to the Pascal VOC metric [8], which determines true and false positives by means of an IoU threshold equal to 0.5 (AP@.5), which is consistent with our definition of meta classification. For the Faster-RCNN we evaluate our method also on these 33 different score thresholds, with one exception which is the Kitti dataset. The classes “person” and “car” are often predicted with extremly confident scores (very close to 1 or 0) such that the predictions differ only marginally at thresholds near 0.8 and 0.1. Therefore, thresholds of $10^{-1}, \dots, 10^{-12}$ were chosen were chosen for the KITTI dataset. This leads to highest obtained mAP values for Faster-RCNN which are 89.33% on Kitti, 80.31% on Pascal VOC and 55.29% on COCO.

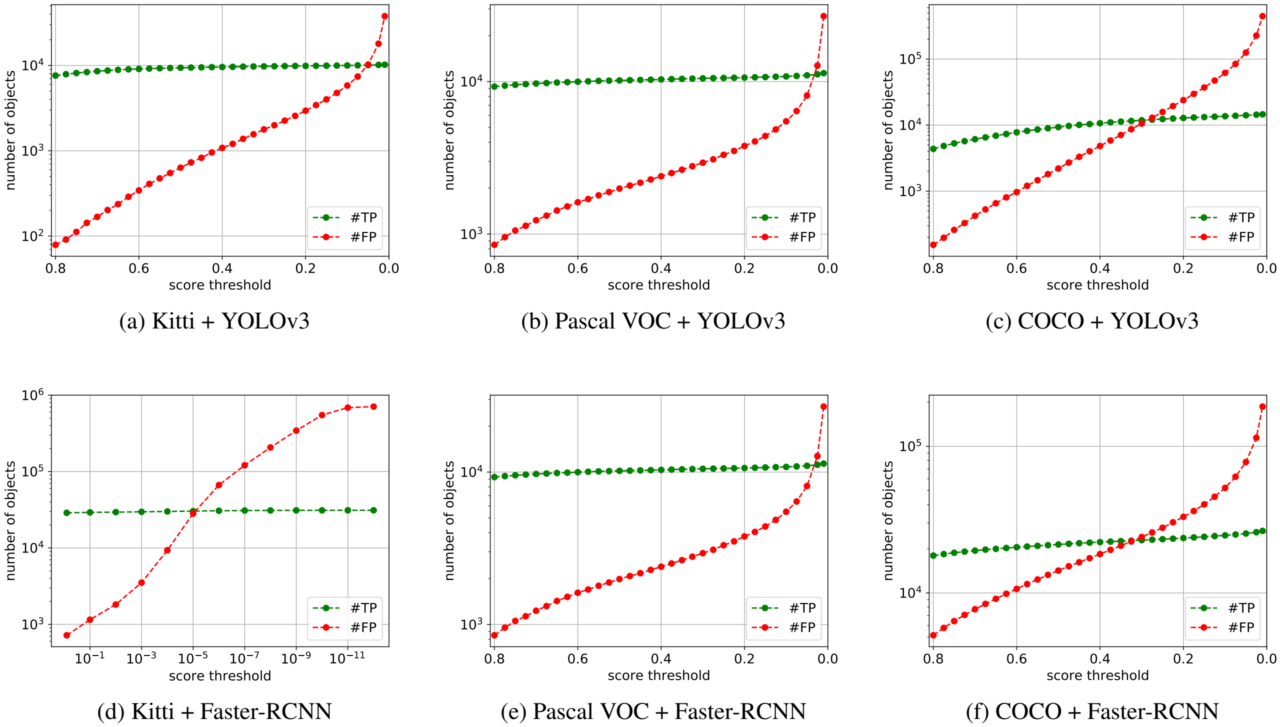
$s^{(i)}_{std}$	0.8370	$s^{(i)}_{mean,MC}$	0.8239
$s^{(i)}$	0.8231	$s^{(i)}_{mean}$	0.8147
$N^{(i)}$	0.7149	$IoU^{(i)}_{pb,std}$	0.6529
$IoU^{(i)}_{pb,high}$	0.6021	$IoU^{(i)}_{pb,mean}$	0.5019
$\hat{r}^{(i)}_{\max,high}$	0.4174	$\hat{r}^{(i)}_{\max}$	0.4054
$d/g^{(i)}_{high}$	0.4049	$\hat{r}^{(i)}_{\max,mean,MC}$	0.4044
$x^{(i)}_{std}$	0.4029	$\hat{r}^{(i)}_{\max,std}$	0.4013
$d/g^{(i)}_{std}$	0.3995	$g^{(i)}_{std}$	0.3904
$\hat{r}^{(i)}_{\max,mean}$	0.3876	$g^{(i)}_{high}$	0.3821

Table 1: Strongest Pearson correlation coefficients for some constructed box-wise metrics for the Kitti dataset, the YOLOv3 network and a score threshold $t = 0.01$.

Figure 2 depicts the number of true positives and false positives for the YOLOv3 and Faster-RCNN network and the Kitti, Pascal VOC and COCO datasets. It is intuitively clear that as the score threshold decreases the number TPs and FPs increases. The sum of TPs and FPs equals the number of all predicted bounding boxes and therefore constitutes the number of examples for training and evaluation of our meta models. The order of magnitude of the number of predictions is between 10^3 and 10^6.

In what follows, all results (if not stated otherwise) were computed from 10 repeated runs where training and validation sets (both of the same size) were re-sampled. We give mean results as well as standard deviations over the obtained results in brackets.

Correlation of box-wise metrics with the IoU. Table 1 contains the Pearson correlation coefficients of the box-wise metrics with the IoU of prediction and ground truth for the Kitti test images and a score threshold $t = 0.01$ for the YOLOv3 network. The score metrics seem to have strong correlations with the IoU, which is expected as it is supposed to discriminate true positives from false positives. The endings *high*, *mean*, *std* represent the maximum, the arithmetic mean and the standard deviation, respectively, of the corresponding filtered candidate boxes for a given metric. The ending *mean*, *MC* represents the arithmetic mean of the corresponding dropout predictions $\hat{b}^{(i)}, \hat{b}^{(i)}_1, ..., \hat{b}^{(i)}_J$ for the respective metric. Note that, although the 4 score related metrics ($s^{(i)}_{std}$, $s^{(i)}_{mean,MC}$, $s^{(i)}$, $s^{(i)}_{mean}$) show the highest correlation with the $IoU \geq 0.8$, these metrics may be very similar. Indeed, the correlation coefficients between these four metrics range from 0.81 to 0.99. Four additional metrics ($N^{(i)}$, $IoU^{(i)}_{pb_{std}}$, $IoU^{(i)}_{pb,high}$, $IoU^{(i)}_{pb,mean}$)



(a) Kitti + YOLOv3

(b) Pascal VOC + YOLOv3

(c) COCO + YOLOv3

(d) Kitti + Faster-RCNN

(e) Pascal VOC + Faster-RCNN

(f) COCO + Faster-RCNN

Figure 2: Comparison of the number of true positives (TP) and false positives (FP) for different score thresholds, the YOLOv3 and Faster-RCNN network and the Kitti, Pascal VOC and COCO datasets.

also show decent correlations $0.5 \leq \rho \leq 0.8$, all other metrics only show a minor correlation. However they may still contribute to a diverse set of metrics.

Comparison of different meta classifiers and regressors. For meta classification (classification of $IoU \geq 0.5$ vs. $IoU < 0.5$), we compare results in terms of classification accuracy and in terms of the area under curve corresponding to the receiver operator characteristic curve (AUROC, see [5]). The receiver operator characteristic curve is obtained by varying the decision threshold of the classification output for deciding whether $IoU \geq 0.5$ or $IoU < 0.5$. For the task of meta regression we state resulting standard deviations σ of the linear regression fit's residual as well as R^2 values. Throughout this section, we consider the following combinations of inputs: Score / score baseline refers to a meta regressor or classifier that was only trained by means of the score metric $s^{(i)}$ which is a sensible baseline. MetaDetect refers to a meta regressor or classifier trained with $46 + C$ metrics that include all metrics except for those that correspond to MC dropout. MetaDetect+Dropout refers to all available $66 + C$ metrics including dropout. MetaDetect+Dropout is only available for YOLOv3, since for this network we inserted a dropout layer with dropout rate 0.5 in all three detection branches right before the final convolutional layer and retrained the network.

For the task of meta regression and meta classification several models (meta models) can be used. The meta models we consider are linear regression (LR), gradient boosting (GB) and a shallow neural network (NN) with two hidden layers. In table 2 we state R^2-values for meta regression with the Kitti dataset and the YOLOv3 network for three different score thresholds t. We be observed that gradient boosting outperforms linear regression and the shallow neural network consistently for all three score thresholds. This is in accordance to the results depicted by fig. 4. Gradient boosting outperforms the linear regression and the shallow neural network as a meta regression model for all 33 different score thresholds. In all our tests, we observed the same behavior for meta regression and meta classification for all three datasets. For this reason, only meta regression and meta classification results with gradient boosting are presented in the following. These findings indicate that there is a significant portion of mutual information contained in our metrics. The increase in R^2 when going from a single score metric to all our $46 + C$ metrics is ranging from 1.52 to 9.83 percent points (pp). On average the gain is about 4.86 pp. When further extending the considered metrics to $66 + C$ by the MC dropout metrics, we do neither observe a significant nor consistent increase in R^2 values.

Meta Regression IoU						
Network	Dataset	Score threshold t	Method	Score baseline	MetaDetect	MetaDetect+Dropout
YOLOv3	Kitti	0.5	LR	$0.4188(\pm 0.0080)$	$0.4447(\pm 0.0059)$	$0.4477(\pm 0.0065)$
			GB	$0.3966(\pm 0.0069)$	$0.4485(\pm 0.0104)$	$0.4478(\pm 0.0102)$
			NN	$0.3448(\pm 0.0096)$	$0.4435(\pm 0.0097)$	$0.4436(\pm 0.0114)$
		0.3	LR	$0.5837(\pm 0.0097)$	$0.6131(\pm 0.0094)$	$0.6116(\pm 0.0098)$
			GB	$0.5679(\pm 0.0099)$	$0.6206(\pm 0.0095)$	$0.6216(\pm 0.0084)$
			NN	$0.5704(\pm 0.0088)$	$0.6078(\pm 0.0094)$	$0.6166(\pm 0.0096)$
		0.1	LR	$0.7133(\pm 0.0032)$	$0.7594(\pm 0.0032)$	$0.7591(\pm 0.0033)$
			GB	$0.7138(\pm 0.0037)$	$0.7766(\pm 0.0025)$	$0.7780(\pm 0.0025)$
			NN	$0.7120(\pm 0.0029)$	$0.7568(\pm 0.0042)$	$0.7635(\pm 0.0060)$

Table 2: Comparison of R^2 values for the score baseline and all available metrics (with and without dropout) for Kitti and YOLOv3. We used linear regression (LR), gradient boosting (GB) and shallow neural nets (NN) for the task of meta regression.

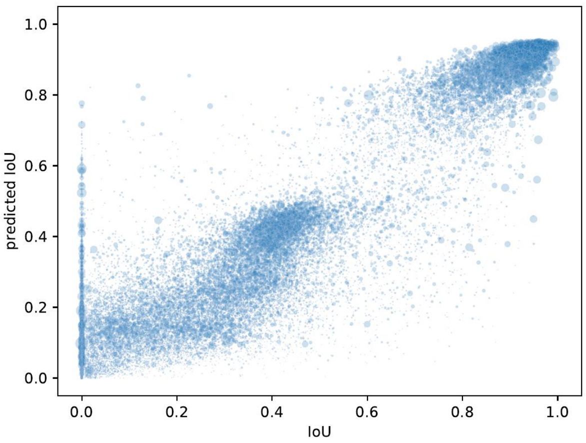


Figure 3: Box-wise scatter plot of true IoU and predicted IoU values for the Kitti dataset, the YOLOv3 network and a score threshold $t = 0.01$. The predicted IoU values are generated with gradient boosting.

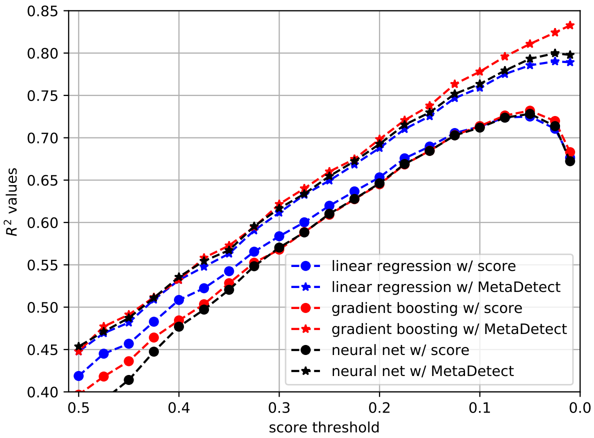


Figure 4: R^2 values for the task of meta regression for the score baseline and all uncertainty measures for the Kitti dataset, the YOLOv3 network and different score values. The R^2 values are calculated with linear regression, gradient boosting and a shallow neural net.

Comparisons for different datasets, networks and sets of metrics. Due to our observations in the previous paragraph, we fix all our meta classifies and regressors to gradient boosting. Table 3 presents results for meta regression in terms of regression R^2 for all three datasets and both networks with three score thresholds each. Obviously, the tendencies indicated by the previously studied table 2 are confirmed by table 3 for the different datasets and networks. In all cases we MetaDetect provides distinct increases in comparison to the score baseline. For the COCO dataset, YOLOv3 network and score threshold 0.1 we even observe an increase of 9.83 percent points. On average, the incease amounts to 4.86 percent points. Analogously to our findings in the previous paragraph, MetaDetect+Dropout is not able to further improve. In our tests we also observed, although the dropout rate is set to 0.5, that the variation introduced by dropout inference seems to be rather limited. We performed 10 forward passes under dropout. In general, a more stochastic behavior of the inference, which could be promoted by additional dropout layers of an analogous batch normalization approach, could lead to an improvement. Also here, we refer to the modular nature of our framework MetaDetect which allows for the incorporation of any uncertainty metric.

Figure 3 shows a scatter plot of the true IoU of prediction and ground truth and the IoU estimated by MetaDetect. The scattered points a well concentrated along the diagonal axis corresponding to the identity $\mathrm{id} : a \mapsto a$. This signals, that we obtain a well calibrated IoU estimate. The limited deviation from the identity shows, that is estimate

Meta Regression IoU					
Network	Dataset	Score threshold t	Score baseline	MetaDetect	MetaDetect+Dropout
YOLOv3	Kitti	0.5	$0.3966(\pm 0.0069)$	$0.4485(\pm 0.0104)$	$0.4478(\pm 0.0102)$
		0.3	$0.5679(\pm 0.0099)$	$0.6206(\pm 0.0095)$	$0.6216(\pm 0.0084)$
		0.1	$0.7138(\pm 0.0037)$	$0.7766(\pm 0.0025)$	$0.7780(\pm 0.0025)$
	Pascal VOC	0.5	$0.3468(\pm 0.0094)$	$0.3853(\pm 0.0113)$	$0.3851(\pm 0.0098)$
		0.3	$0.4234(\pm 0.0081)$	$0.4700(\pm 0.0078)$	$0.4672(\pm 0.0089)$
		0.1	$0.5384(\pm 0.0051)$	$0.5907(\pm 0.0057)$	$0.5879(\pm 0.0065)$
	COCO	0.5	$0.2317(\pm 0.0089)$	$0.2787(\pm 0.0077)$	$0.2774(\pm 0.0084)$
		0.3	$0.4428(\pm 0.0058)$	$0.4978(\pm 0.0070)$	$0.4960(\pm 0.0062)$
		0.1	$0.4931(\pm 0.0028)$	$0.5914(\pm 0.0030)$	$0.5911(\pm 0.0028)$
Faster-RCNN	Kitti	10^{-1}	$0.3703(\pm 0.0136)$	$0.4101(\pm 0.0153)$	
		10^{-6}	$0.8930(\pm 0.0017)$	$0.9178(\pm 0.0014)$	
		10^{-12}	$0.7927(\pm 0.0020)$	$0.8819(\pm 0.0013)$	
	Pascal VOC	0.5	$0.4844(\pm 0.0106)$	$0.5430(\pm 0.0121)$	
		0.3	$0.5682(\pm 0.0084)$	$0.6182(\pm 0.0089)$	
		0.1	$0.6289(\pm 0.0054)$	$0.6836(\pm 0.0074)$	
	COCO	0.5	$0.4178(\pm 0.0046)$	$0.4330(\pm 0.0052)$	
		0.3	$0.4730(\pm 0.0076)$	$0.4890(\pm 0.0068)$	
		0.1	$0.5438(\pm 0.0056)$	$0.5651(\pm 0.0041)$	

Table 3: Comparison of R^2 values for the score baseline and all available metrics (with and without dropout) for Kitti, Pascal VOC and COCO. Especially, for Faster-RCNN there are no dropout metrics available. We used gradient boosting (GB) for the task of meta regression.

gives rise to predictive uncertainty and an object-wise quality estimate. For example images, we refer to fig. 1

Considering the task of meta classification which amounts to false positive detection, we also achieve clear improvements when considering all our metrics instead of only considering the score $s^{(i)}$. In this task we can also consider another baseline which is given just by random guessing. In case of a balanced dataset with equal amount of false positives and false negatives, both the classification accuracy and the AUROC have values of 0.5 in the case of random guessing.

Results are summarized in table 4 in terms of classification accuracy and in table 5 in terms of classification AUROC. Since different score thresholds t amount to different ratios of false positives and true positives, we apply SMOTE [3] in order to balance the meta classification dataset. Our findings in these tables are similar to those for meta regression, the advantage of MetaDetect over the score baseline seems to be slightly more pronounced in terms of AUROC than in terms of accuracy. In terms of accuracy, we observe an average increase of 2.07 pp, while in terms of AUROC, we obtain an average increase of 2.32 pp. The highest increase overall is obtained for the YOLOv3 network and the COCO dataset with a score threshold $t = 0.5$ with 7.09 pp in terms of classification accuracy and for the YOLOv3 network and the COCO dataset with a score threshold $t = 0.5$ with 9.18 pp in terms of AUROC.

Figure 5 shows examples of predicted bounding boxes that have a true $IoU = 0$. Thus, according to the ground truth they are FPs, but with high meta classification probabilities. Indeed, these boxes each represent an object of the correct class. Hence, more reliable meta classifiers may also help to identify labeling errors.

5. Conclusion and Outlook

In this work we extended the notion of [17] from classification to object detection, considering the tasks of meta classification and meta regression introduced in [34, 36, 25] for semantic segmentation. We introduced a generic framework for quality estimation and false positive detection that can be extended by any uncertainty measure for object detection. To demonstrate the ability of our framework to perform these tasks more efficiently than state-of-the-art baselines, we introduced a variety of different uncertainty metrics and also considered widely used MC dropout [11] uncertainty. To study the individual metric's performance we compared correlation coefficients of the different metrics with each other and found that some of our metrics may well contribute to the overall meta regression and classification performance. This is confirmed by further numerical experiments where we compared different meta regressors and classifiers, i.e., logistic/linear regression, gradient boosting and shallow neural networks. We found that gradient boosting yields the best performance. Furthermore, with a thorough study of meta classification and regression performance over three different datasets and two different

Meta Classification $IoU \geq 0.5$ vs. $IoU < 0.5$, Accuracies					
Network	Dataset	Score threshold t	Score baseline	MetaDetect	MetaDetect+Dropout
YOLOv3	Kitti	0.5	$0.8527(\pm 0.0132)$	$0.9042(\pm 0.0145)$	$0.9032(\pm 0.0129)$
		0.3	$0.8905(\pm 0.0075)$	$0.9092(\pm 0.0100)$	$0.9090(\pm 0.0091)$
		0.1	$0.9167(\pm 0.0050)$	$0.9207(\pm 0.0058)$	$0.9205(\pm 0.0054)$
	Pascal VOC	0.5	$0.8106(\pm 0.0097)$	$0.8527(\pm 0.0077)$	$0.8537(\pm 0.0072)$
		0.3	$0.8259(\pm 0.0077)$	$0.8532(\pm 0.0104)$	$0.8524(\pm 0.0103)$
		0.1	$0.8480(\pm 0.0040)$	$0.8628(\pm 0.0048)$	$0.8624(\pm 0.0047)$
	COCO	0.5	$0.7381(\pm 0.0118)$	$0.8090(\pm 0.0112)$	$0.8083(\pm 0.0115)$
		0.3	$0.7882(\pm 0.0066)$	$0.8022(\pm 0.0070)$	$0.8023(\pm 0.0069)$
		0.1	$0.8400(\pm 0.0032)$	$0.8861(\pm 0.0036)$	$0.8854(\pm 0.0036)$
Faster-RCNN	Kitti	10^{-1}	$0.9092(\pm 0.0062)$	$0.9266(\pm 0.0060)$	
		10^{-6}	$0.9678(\pm 0.0017)$	$0.9710(\pm 0.0013)$	
		10^{-12}	$0.9825(\pm 0.0023)$	$0.9892(\pm 0.0004)$	
	Pascal VOC	0.5	$0.8270(\pm 0.0043)$	$0.8384(\pm 0.0049)$	
		0.3	$0.8420(\pm 0.0042)$	$0.8508(\pm 0.0056)$	
		0.1	$0.8642(\pm 0.0040)$	$0.8726(\pm 0.0043)$	
	COCO	0.5	$0.7814(\pm 0.0035)$	$0.7865(\pm 0.0036)$	
		0.3	$0.7877(\pm 0.0041)$	$0.7913(\pm 0.0029)$	
		0.1	$0.8246(\pm 0.0027)$	$0.8430(\pm 0.0032)$	

Table 4: Comparison of meta classification accuracies for the score baseline and all available metrics (with and without dropout) for the Kitti, Pascal VOC and COCO datasets. For Faster-RCNN there are no dropout metrics available. We used gradient boosting (GB) for meta classification.

Meta Classification $IoU \geq 0.5$ vs. $IoU < 0.5$, AUROCs					
Network	Dataset	Score threshold t	Score baseline	MetaDetect	MetaDetect+Dropout
YOLOv3	Kitti	0.5	$0.9183(\pm 0.0090)$	$0.9635(\pm 0.0065)$	$0.9626(\pm 0.0072)$
		0.3	$0.9439(\pm 0.0058)$	$0.9655(\pm 0.0049)$	$0.9651(\pm 0.0049)$
		0.1	$0.9598(\pm 0.0031)$	$0.9717(\pm 0.0025)$	$0.9714(\pm 0.0023)$
	Pascal VOC	0.5	$0.8772(\pm 0.0082)$	$0.9306(\pm 0.0049)$	$0.9304(\pm 0.0040)$
		0.3	$0.8950(\pm 0.0072)$	$0.9266(\pm 0.0061)$	$0.9265(\pm 0.0057)$
		0.1	$0.9183(\pm 0.0035)$	$0.9327(\pm 0.0030)$	$0.9326(\pm 0.0027)$
	COCO	0.5	$0.8063(\pm 0.0145)$	$0.8981(\pm 0.0106)$	$0.8979(\pm 0.0098)$
		0.3	$0.8578(\pm 0.0070)$	$0.8793(\pm 0.0062)$	$0.8793(\pm 0.0062)$
		0.1	$0.9125(\pm 0.0027)$	$0.9514(\pm 0.0025)$	$0.9515(\pm 0.0020)$
Faster-RCNN	Kitti	10^{-1}	$0.9580(\pm 0.0051)$	$0.9694(\pm 0.0036)$	
		10^{-6}	$0.9891(\pm 0.0009)$	$0.9949(\pm 0.0004)$	
		10^{-12}	$0.9982(\pm 0.0002)$	$0.9993(\pm 0.0001)$	
	Pascal VOC	0.5	$0.8930(\pm 0.0036)$	$0.9090(\pm 0.0036)$	
		0.3	$0.9113(\pm 0.0036)$	$0.9196(\pm 0.0034)$	
		0.1	$0.9309(\pm 0.0028)$	$0.9399(\pm 0.0028)$	
	COCO	0.5	$0.8482(\pm 0.0022)$	$0.8608(\pm 0.0032)$	
		0.3	$0.8623(\pm 0.0033)$	$0.8669(\pm 0.0028)$	
		0.1	$0.8979(\pm 0.0021)$	$0.9157(\pm 0.0022)$	

Table 5: Comparison of Meta-Classification-AUROC values for the score baseline and all available metrics (with and without dropout) for the Kitti, Pascal VOC and COCO datasets. For Faster-RCNN there are no dropout metrics available since the pre-trained network does not use dropout. We used gradient boosting (GB) for meta classification.

object detection networks, our method consistently outperforms the score baseline by a significant margin. In terms of meta classification we improve over the results of the score baseline by up to 7.09 pp classification accuracy and 9.18 pp AUROC. For meta regression obtain an improvement up to 9.83 pp in R^2.

We plan to incorporate these improved quality estimates into an active learning pipeline as well as performing an

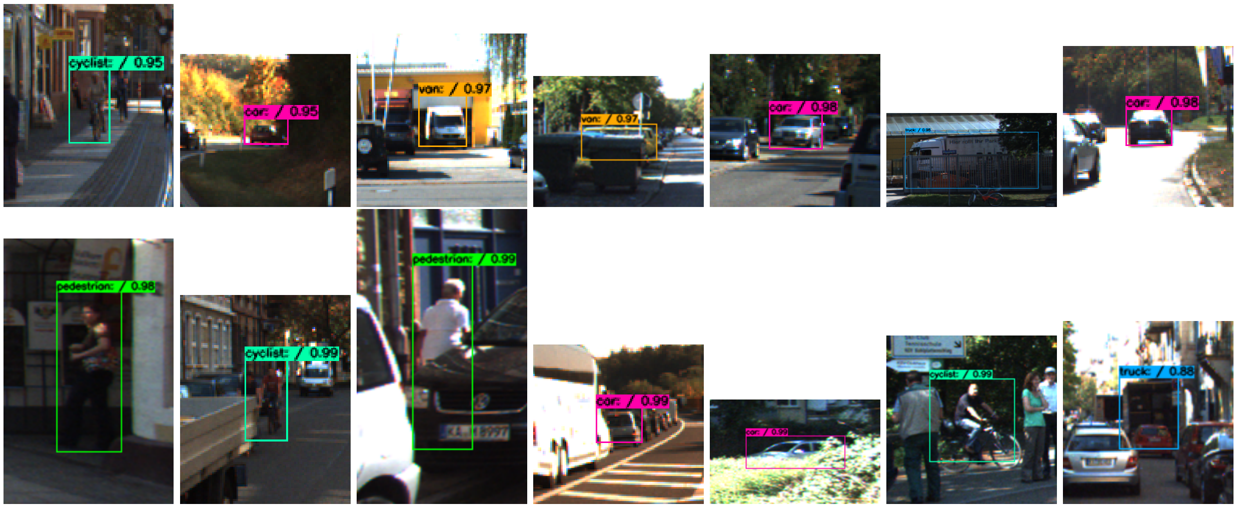


Figure 5: Examples of predicted bounding boxes with a true $IoU = 0$ but with high meta classification probabilities. These examples predicted by the YOLOv3 network are counted as FPs as there is no corresponding ground truth available. Our prediction quality estimates signal with very high IoU values, that an object is present. The predictions, and therefore the data to train and evaluate gradient boosting, are obtained by the YOLOv3 network applied to the Kitti dataset.

evaluation of the applicability of MetaDetect to data quality estimation. We believe that many labeling erros can be detected by means of a well-calibrated quality estimate.

We make our source code publicly available at https://github.com/schubertm/MetaDetect.

Acknowledgements. We thank Hanno Gottschalk for discussion and useful advice. Furthermore, we acknowledge support by the European Regional Development Fund (ERDF), grant-no. EFRE-0400216.

EUROPEAN UNION
Investing in our Future
European Regional
Development Fund

References

[1] M. Abadi, A. Agarwal, P. Barham, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] H. Attias. A variational bayesian framework for graphical models. In *In Advances in Neural Information Processing Systems 12*, pages 209–215. MIT Press, 2000.

[3] K. W. Bowyer, N. V. Chawla, L. O. Hall, and W. P. Kegelmeyer. SMOTE: synthetic minority over-sampling technique. *CoRR*, abs/1106.1813, 2011.

[4] R. Chan, M. Rottmann, F. Hüger, P. Schlicht, and H. Gottschalk. Metafusion: Controlled false-negative reduction of minority classes in semantic segmentation. *ArXiv*, abs/1912.07420, 2019.

[5] J. Davis and M. Goadrich. The relationship between precision-recall and ROC curves. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, pages 233–240, 2006.

[6] D. Duvenaud, D. Maclaurin, and R. P. Adams. Early stopping as nonparametric variational inference. *ArXiv*, abs/1504.01344, 2016.

[7] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, Jan. 2015.

[8] M. Everingham, L. Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vision*, 88(2):303–338, June 2010.

[9] D. Feng, L. Rosenbaum, and K. Dietmayer. Towards safe autonomous driving: Capture uncertainty in the deep neural network for lidar 3d vehicle detection. *CoRR*, abs/1804.05132, 2018.

[10] D. Feng, L. Rosenbaum, F. Timm, and K. Dietmayer. Leveraging heteroscedastic aleatoric uncertainties for robust real-time lidar 3d object detection. *CoRR*, abs/1809.05590, 2018.

[11] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In M. F. Balcan and K. Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR.

[12] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[13] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint*, 2014.

[14] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On calibration of modern neural networks. *ArXiv*, abs/1706.04599, 2017.

[15] A. Harakeh, M. Smart, and S. L. Waslander. Bayesod: A bayesian approach for uncertainty estimation in deep object detectors. *CoRR*, abs/1903.03838, 2019.

[16] Y. He, X. Zhang, M. Savvides, and K. Kitani. Softer-nms: Rethinking bounding box regression for accurate object detection. *CoRR*, abs/1809.08545, 2018.

[17] D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. *CoRR*, abs/1610.02136, 2016.

[18] C. Huang, Q. Wu, and F. Meng. Qualitynet: Segmentation quality evaluation with deep convolutional networks. In *2016 Visual Communications and Image Processing (VCIP)*, pages 1–4, Nov 2016.

[19] E. Hüllermeier and W. Waegeman. Aleatoric and epistemic uncertainty in machine learning: A tutorial introduction. 10 2019.

[20] F. Kraus and K. Dietmayer. Uncertainty estimation in one-stage object detection. *CoRR*, abs/1905.10296, 2019.

[21] F. Küppers, J. Kronenberger, A. Shantia, and nselm Haselhoff. Multivariate confidence calibration for object detection. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1322–1330, 2020.

[22] M. Le, F. Diehl, T. Brunner, and A. Knol. Uncertainty estimation for deep neural object detectors in safety-critical applications. pages 3873–3878, 11 2018.

[23] S. Liang, Y. Li, and R. Srikant. Principled detection of out-of-distribution examples in neural networks. 06 2017.

[24] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014.

[25] K. Maag, M. Rottmann, and H. Gottschalk. Time-dynamic estimates of the reliability of deep semantic segmentation networks. *CoRR*, abs/1911.05075, 2019.

[26] D. J. C. MacKay. A practical bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992.

[27] D. Miller, F. Dayoub, M. Milford, and N. Sünderhauf. Evaluating merging strategies for sampling-based uncertainty techniques in object detection. *CoRR*, abs/1809.06006, 2018.

[28] D. Miller, L. Nicholson, F. Dayoub, and N. Sünderhauf. Dropout sampling for robust object detection in open-set conditions. *CoRR*, abs/1710.06677, 2017.

[29] O. Ozdemir, B. Woodward, and A. A. Berlin. Propagating uncertainty in multi-stage bayesian convolutional neural networks with application to pulmonary nodule detection. *CoRR*, abs/1712.00497, 2017.

[30] B. Phan, R. Salay, K. Czarnecki, V. Abdelzad, T. Denouden, and S. Vernekar. Calibrating uncertainties in object localization task. *CoRR*, abs/1811.11210, 2018.

[31] J. Redmon, S. K. Divvala, R. B. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *CoRR*, abs/1506.02640, 2015.

[32] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018.

[33] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[34] M. Rottmann, P. Colling, T. Hack, F. Hüger, P. Schlicht, and H. Gottschalk. Prediction error meta classification in semantic segmentation: Detection via aggregated dispersion measures of softmax probabilities. *CoRR*, abs/1811.00648, 2018.

[35] M. Rottmann, K. Maag, R. Chan, F. Hüger, P. Schlicht, and H. Gottschalk. Detection of false positive and false negative samples in semantic segmentation. In *2020 Design, Automation & Test in Europe Conference & Exhibition, DATE 2020, Grenoble, France, March 9-13, 2020*, pages 1351–1356. IEEE, 2020.

[36] M. Rottmann and M. Schubert. Uncertainty measures and prediction quality rating for the semantic segmentation of nested multi resolution street scene images. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops, CVPR Workshops 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 1361–1369. Computer Vision Foundation / IEEE, 2019.

[37] A. G. Roy, S. Conjeti, N. Navab, and C. Wachinger. Inherent brain segmentation quality control from fully convnet monte carlo sampling. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 664–672. Springer, 2018.