K. Maag, M. Rottmann, F. Hüger, P. Schlicht and H. Gottschalk

# Improving Video Instance Segmentation by Light-weight Temporal Uncertainty Estimates

December 14, 2020

# Improving Video Instance Segmentation by Light-weight Temporal Uncertainty Estimates

Kira Maag
*University of Wuppertal, Germany*
kmaag@uni-wuppertal.de

Matthias Rottmann
*University of Wuppertal, Germany*
rottmann@uni-wuppertal.de

Fabian Hüger
*Volkswagen Group Automation*
fabian.hueger@volkswagen.de

Peter Schlicht
*Volkswagen Group Automation*
peter.schlicht@volkswagen.de

Hanno Gottschalk
*University of Wuppertal, Germany*
hgottsch@uni-wuppertal.de

***Abstract*—Instance segmentation with neural networks is an essential task in environment perception. However, the networks can predict false positive instances with high confidence values and true positives with low ones. Hence, it is important to accurately model the uncertainties of neural networks to prevent safety issues and foster interpretability. In applications such as automated driving the detection of road users like vehicles and pedestrians is of highest interest. We present a temporal approach to detect false positives and investigate uncertainties of instance segmentation networks. Since image sequences are available for online applications, we track instances over multiple frames and create temporal instance-wise aggregated metrics of uncertainty. The prediction quality is estimated by predicting the intersection over union as performance measure. Furthermore, we show how to use uncertainty information to replace the traditional score value from object detection and improve the overall performance of instance segmentation networks.**

## I. Introduction

Object detection describes the task of identifying and localizing objects of a set of given classes. With respect to image data, state-of-the-art approaches are mostly based on convolutional neural networks (CNNs). Localization can be performed for example by predicting bounding boxes or labeling each pixel that corresponds to a given instance. The latter is also known as instance segmentation (see bottom image in fig. 1) which is an essential tool for scene understanding and considered throughout this work. Prediction quality estimates [1]–[3] as well as uncertainty quantification [4] of neural networks are of highest interest in safety critical applications like medical diagnosis [5] and automated driving [6]. However, instance segmentation networks such as YOLACT [7] and Mask R-CNN [8] do not give well adjusted uncertainty estimations [9]. These networks provide a confidence value, also called score value, for each instance which can have high values for false predictions and low ones for correct predictions. Confidence calibration addresses this problem [9]. In object detection, the confidence values are adjusted to reduce the error between the average precision (as a performance measure) and the confidence values [10]. During inference of instance segmentation networks, all instances with score values below a threshold are removed. It can happen that correctly predicted instances disappear as well as many false positives remain. For this reason, we do not use a score threshold during inference and present an uncertainty quantification method that gives more accurate information compared to the score value. We utilize this uncertainty quantification to improve the networks' performance in terms of accuracy. Another approach to improve the trade-off between false negatives and false positives in semantic segmentation has been introduced in [11]. This work considers the difference in Maximum Likelihood and Bayes decision rule.

This work is supported by Volkswagen Group Automation.

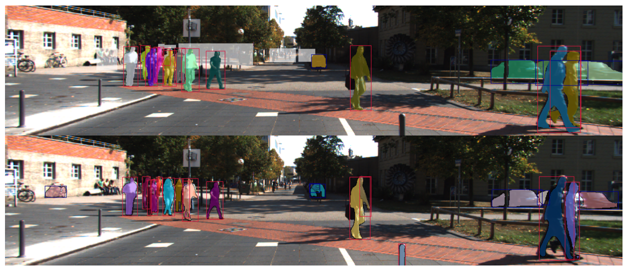
Fig. 1. Ground truth image (*top*) with ignored regions (white) and instance segmentation (*bottom*). The bounding boxes drawn around the instances represent the class, blue denotes cars and red pedestrians.

In this work, we introduce the tasks of *meta classification* and *meta regression* for instance segmentation which was introduced for semantic segmentation in [3]. They provide a framework for post processing a semantic segmentation in order to estimate the quality of each predicted segment. More precisely, meta classification refers to the task of predicting whether a predicted instance intersects with the ground truth or not. A commonly used performance measure is the intersection over union ($IoU$) which quantifies the degree of overlap of prediction and ground truth [12]. In instance segmentation, an object is called false positive if the $IoU$ is less than 0.5. Hence, we consider the task of (meta) classifying between $IoU < 0.5$ and $IoU \geq 0.5$ for every predicted instance. We use meta classification to identify false positive instances and improve the overall network performance compared to the application of a score threshold during inference. The task of meta regression is the prediction of the $IoU$ for each predicted instance directly. Both meta classification and regression (*meta tasks*) are able to reliably evaluate the quality of an instance segmentation obtained from a neural network. In addition, the prediction of the $IoU$ serves as a performance estimate. For learning both meta tasks, we use instance-wise metrics as input for the respective models. In [2] single frame metrics are introduced which characterize uncertainty and geometry of a given semantic segment. By tracking these segments over time, time series of single frame metrics are generated. We apply these metrics to instance segmentation and extend them by a number of new truly time-dynamic metrics. These metrics are based on survival time analysis as well as on changes in the shape and expected position of instances in an image sequence. From this information, we estimate the prediction quality on instance-level by means of temporal uncertainties. In addition, for generating time series we propose a light-weight tracking approach for predicted instances. Our tracking algorithm matches instances according to their overlap in consecutive frames by shifting instances according to their expected location in the subsequent frame.

In this work, we present post processing methods for uncertainty quantification and performance improvement. We only assume that a trained instance segmentation network and image sequences of input data are available. In our tests, we employ two publicly available networks, the YOLACT and the Mask R-CNN network. We apply these networks to the KITTI [13] and the MOT [14] dataset for multi-object tracking and instance segmentation. The source code of our method is publicly available at https://github.com/kmaag/Temporal-Uncertainty-Estimates. Our contributions are summarized as follows:

- We present a light-weight tracking algorithm for instances predicted by a neural network and resulting time-dynamic metrics. These metrics serve as input for different models for meta classification and regression.
- We evaluate our tracking algorithm on the KITTI and the MOT dataset.
- We perform meta classification and regression to evaluate the quality of two instance segmentation networks. Furthermore, we study different types of models for meta tasks w.r.t. their dependence on the length of time series and compare them with different baselines. For meta regression we obtain $R^2$ values of up to $86.85\%$ and for meta classification AUROC values of up to $98.78\%$ which is clearly superior to the performance of previous approaches.
- We compare the meta classification performance with the application of a score threshold during inference. Meta classification reduces the number of false positives by up to $44.03\%$ while maintaining the number of false negatives.

## II. Related Work

*a) False Positive Detection and Uncertainty Quantification:* Bayesian models are one possibility to consider model uncertainty [15]. A well-known approximation to Bayesian inference is the Monte-Carlo (MC) Dropout [4] which has proven to be practically efficient in detecting uncertainties and has also been applied to semantic segmentation tasks [16]. In [17] MC Dropout is also used to filter out predictions with low reliability. To detect spatial and temporal uncertainty, this line of research is further developed in [18]. Based on MC Dropout, structure-wise metrics are presented in [19], as well as voxel-wise uncertainty metrics based on softmax maximum probability in [20]. In [3] the concepts of meta classification and meta regression are introduced with segment-wise metrics as input extracted from the segmentation network's softmax output. This idea is extended in [21] to object detection, in [22] by adding resolution dependent uncertainty and in [2] by a temporal component. Similar works on a single object per image basis are introduced in [1] and [23], instead of hand crafted metrics they utilize additional CNNs. Some methods for uncertainty evaluation are transferred to object detection, such as the MC Dropout [24]. False positive detection is presented in [5] based on MC Dropout and an ensemble approach. In [6] false positive objects obtain high uncertainties using two methods, loss attenuation and redundancy with multi-box detection. Based on a dropout sampling approach for object detection [25], the work presented in [26] investigates the semantic and spatial uncertainty in instance segmentation.

While most works are based on MC Dropout [4], [16]–[18], we apply a post processing method. We use information of the network output, as it is also pursued in confidence calibration. In confidence calibration only score values and box positions are considered to adjust the score values with respect to performance [10]. The work closest to ours is [2] which introduces time-dynamic meta classification and regression for semantic segmentation. In this work, we introduce for the first time a temporal approach to meta classification and regression for instance segmentation. While [2] only provides time series of single frame metrics, we go beyond that and introduce truly time-dynamic metrics that quantify temporal changes in geometry and expected position complemented with quantities derived by a survival analysis. Also for the first time, we demonstrate successfully that time-dynamic meta classification performance can be traded for instance segmentation performance.

*b) Multi-Object Tracking:* Tracking multiple objects in videos, in applications like automated driving, is an essential task in computer vision [14]. The tracking task (association problem) in [27], [28] and [29] is solved by dual matching attention networks, a CNN using quadruplet losses and a CNN based on correlation filters, respectively. The previously described algorithms use one model for the object detection and another for the association problem. In [30] a shared model for both task is presented. A focus on the improvement of long-term appearance models is demonstrated in [31] based on a recurrent network. Most works, including those described here, work with bounding boxes, while there are other methods using a binary segmentation mask representation of the object [32]. In [33] and [34] segmentation and tracking are jointly solved using a pixel-level probability model and a recurrent fully convolutional network, respectively. To perform the detection, segmentation and tracking tasks simultaneously, the Mask R-CNN network is extended by a tracking branch [35], by 3D convolutions to incorporate temporal information [36] and by a mask propagation branch [37]. In contrast, the subproblems classification, detection, segmentation and tracking are treated independently in [38]. Another work for multi-object tracking is based on the optical flow and the Hungarian algorithm [39]. The method introduced in [2] works with semantic segments and is based on the overlap of segments in consecutive frames.

Machine learning is widely used in object tracking approaches as described above, while our tracking method is solely based on the degree of overlap of predicted instances. In the style of the tracking algorithm for semantic segments [2], we introduce a method to track instances. In contrast, we are able to get away with an easy algorithm and less matching steps. For example, segments of the same class that are close to each other are merged to one segment, while this contradicts the idea of instance segmentation. Also due to the different nature of instances and segments, all steps that have a similar intention as in [2] are indeed constructed differently. Due to the lack of data in semantic segmentation, the tracking performance was not evaluated in [2]. We evaluate our tracking algorithm and compare it with the deep learning approach presented in [36]. Note that, in contrast to tracking methods integrated into object detection or instance segmentation networks, our approach is independent of the network and serves as a post processing step.

## III. Temporal Prediction Quality

Instance segmentation is an extension of object detection. In both tasks, multiple bounding boxes with corresponding class affiliations are predicted. In instance segmentation an additional pixel-wise mask representation is included. Also in both tasks, first a score threshold is applied to remove objects with low scores, thereafter a non-maximum suppression is applied to avoid multiple predictions for the same object. Our method is based on temporal information of these remaining instances. We track instances over multiple frames in video sequences and generate time-dynamic metrics. From this information we estimate the prediction quality (meta regression) on instance-level and predict false positive instances (meta classification), also in order to improve the networks' performance in terms of accuracy. In this section, we describe our tracking approach, the temporal metrics as well as the methods used for meta classification and regression.

### A. *Tracking Method for Instances*

In this section, we introduce a light-weight tracking algorithm for predicted instances in image sequences where instance segmentation is available for each frame. As our method is a post processing step, the tracking algorithm is independent of the choice of instance segmentation network. Each instance $i$ of an image $x$ has a label $y$ from a prescribed label space $\mathcal{C}$. In fig. 1 (top) a ground truth image is shown. Therein, the white areas are ignored regions with unlabeled cars and pedestrians. An evaluation for predicted instances in these regions is not possible, therefore all instances where $80\%$ of their number of pixels are inside an ignored region are not considered for tracking and further experiments. Given an image $x$, $\hat{\mathcal{I}}_x$ denotes the set of predicted instances not covered by ignored regions. We match instances of the same class in consecutive frames as follows: Instances are matched according to their overlap or if their geometric centers are close. For this purpose, we shift instances according to their expected location in the next frame using information from previous frames. Additionally, we use a linear regression to match instances that are at least one and at most $t_l - 2$ frames apart in temporal direction in order to account for flashing predicted instances (temporary false negatives or occluded instances). We define the overlap of two instances $i$ and $j$ by

$$O_{i,j} = \frac{|i \cap j|}{|i \cup j|} \tag{1}$$

and the geometric center of instance $i$ in frame $t$ by

$$\bar{i}_t = \frac{1}{|i|} \sum_{(z_v, z_h) \in i} (z_v, z_h) \tag{2}$$

where $(z_v, z_h)$ describes the vertical and horizontal coordinate of pixel $z$. We denote by $\{x_t : t = 1, \ldots, T\}$ the image sequence with a length of $T$. Our tracking algorithm is applied sequentially to each frame $t = 1, \ldots, T$. The instances in frame 1 are assigned with random ids. Then, the ones in frame $t-1$ follow a tracking procedure to match with instances in frame $t$. To give priorities for matching, the instances are sorted by size and passed in descending order. A precise description of how an instance $i \in \hat{\mathcal{I}}_{x_{t-1}}$ in frame $t-1$ is matched with an instance $j \in \hat{\mathcal{I}}_{x_t}$ in frame $t$ is described in algo. 1. This is performed for every instance $i$. If the instances $i$ and $j$ are matched, the algorithm terminates and instance $j$ is excluded from further considerations in order to preserve the id's uniqueness. When the algorithm has visited all instances $i \in \hat{\mathcal{I}}_{x_{t-1}}$, the instances $j \in \hat{\mathcal{I}}_{x_t}$ that have not been matched are assigned with new ids. Within the description of the tracking algorithm, we introduce parameters $c_o$, $c_d$ and $c_l$ as thresholds for distance, overlap and distance after shifting according to linear regression, respectively.

### B. *Temporal Instance-wise Metrics*

First, we consider the metrics introduced in [2] applied to instances. These metrics are single frame metrics based on an object's geometry, like instance size and geometric center, as well as extracted from the segmentation network's softmax output, such as average over an instance's pixel-wise entropy. In general, to calculate instance-wise metrics

**Algorithm 1:** Tracking algorithm

```
/* shift */
if $t > 2$ and instance $i$ exists in frame $t-2$ then
    shift instance $i$ from frame $t-1$ by the vector
      $(\bar{i}_{t-1} - \bar{i}_{t-2})$
    if $\max_{j \in \hat{\mathcal{I}}_{x_t}} O_{i,j} \geq c_o$ then
        match instances $i$ and $j$
    else if $\min_{j \in \hat{\mathcal{I}}_{x_t}} \|\bar{j}_t - \bar{i}_{t-1}\|_2 +$
      $\|(\bar{i}_{t-1} - \bar{i}_{t-2}) - (\bar{j}_t - \bar{i}_{t-1})\|_2 \leq c_d$ then
        match instances $i$ and $j$
/* distance */
if $t > 1$ and $i$ does not exist in frame $t-2$ then
    if $\min_{j \in \hat{\mathcal{I}}_{x_t}} \|\bar{j}_t - \bar{i}_{t-1}\|_2 \leq c_d$ then
        match instances $i$ and $j$
/* overlap */
if $t > 1$ and $\max_{j \in \hat{\mathcal{I}}_{x_t}} O_{i,j} \geq c_o$ then
    match instances $i$ and $j$
/* regression */
if $t > 3$ and instance $i$ appears in at least two of the
  frames $t - t_l, \ldots, t-1$ then
    compute geometric centers of instance $i$ in frames
      $t - t_l$ to $t-1$ (in case $i$ exists in all these frames)
    perform linear regression to predict the geometric
      center $(\hat{\bar{i}}_t)$
    if $\min_{j \in \hat{\mathcal{I}}_{x_t}} \left\|\bar{j}_t - \hat{\bar{i}}_t\right\|_2 \leq c_l$ then
        match instances $i$ and $j$
    else
        shift instance $i \in \hat{\mathcal{I}}_{x_{t_{max}}}$ by the vector
          $\left(\hat{\bar{i}}_t - \bar{i}_{t_{max}}\right)$ where $t_{max} \in \{t - t_l, \ldots, t-1\}$
          denotes the frame where $i$ contains the
          maximum number of pixels
        if $\max_{j \in \hat{\mathcal{I}}_{x_t}} O_{i,j} \geq c_o$ then
            match instances $i$ and $j$
```

from any uncertainty heatmap (like pixel-wise entropy), we compute the mean of the pixel-wise uncertainty values of a given instance. In addition, an instance is divided into inner and boundary. The ratio of pixels in the inner and the boundary indicates fractal shaped instances which signals a false prediction. Moreover, we analogously define uncertainty metrics for the inner and boundary since uncertainties may be higher on an instance's boundary. To this end, for each pixel $z$ a probability distribution over the classes $y$ is required. If the network does not provide a probability distribution for each pixel, but only for the instance, we only use those metrics of [2] that can be computed from the predicted instance mask. We denote both sets of metrics by $U^i$ independent of the network. These metrics serve as a baseline in our tests and as a basis for the following metrics.

Next, we define additional metrics mostly based on temporal information extracted by the tracking algorithm. Instance segmentation as an extension of object detection provides for each instance a confidence value. We add this score, denoted by $s$, to our set of metrics. The next metric is based on the variation of instances in consecutive frames. Instance $i$ of frame $t-1$ is shifted such that $i$ and its matched counterpart $j$ in frame $t$ have a common geometric center. We then calculate the overlap (see eq. (1)) as a measure of shape preservation $f$. Large deformations may indicate poorly predicted instances. In the following, time series of the previously shown metrics are constructed. For each instance $i$ in frame $t$ we gather a time series of the geometric centers $\bar{i}_k$, $k = t-5, \ldots, t-1$. If the instance exists in at least two previous frames, the geometric center $\hat{\bar{i}}_t$ in frame $t$ is predicted using linear regression. The deviation between geometric center and expected geometric center $\|\hat{\bar{i}}_t - \bar{i}_t\|_2$ is used as a time-dynamic measure denoted by $d_c$. We proceed analogously with the instance size $S = |i|$ and compute the deviation $|\hat{S} - S| =: d_s$. Small deviations $d_c$ and $d_s$ indicate that the predicted instance is consistent over time. The following metric is based on a survival analysis [40] of the instances. We choose all predicted instances that are matched in the previous five frames with the same ground truth instance. A predicted instance $i$ and a ground truth instance $g$ are considered as a match if they have an $IoU \geq 0.5$. The associated survival time of instance $i$ in frame $t$ is described by the number of frames in which ground truth instance $g$ appears in consecutive frames. By the proposed tracking method, we obtain time series for each of the previously presented single-frame metrics $U_k^i \cup \{s_k\}$, $k = t-5, \ldots, t$. These serve as input for a Cox regression [41] survival model which predicts the survival times $v$ of instances. Long survival times indicate reliable instances, while very short survival times suggest uncertainty. The last measure is based on the height to width ratio of the instances. To this end, we calculate the average height to width ratio of ground truth instances $g_c$ per class $c$ by

$$r_{gt,c} = \frac{1}{|g_c|} \sum_{g=1}^{g_c} \frac{h_g}{w_g} \tag{3}$$

where $h_g$ denotes the height and $w_g$ the width of an instance $g$. We separate the ratio by class, as for instance cars and pedestrians typically have different ratios of height and width. False predictions can result in deviations of these typical ratios. The ratio metric for a predicted instance $i$ is given by $r := (h_i/w_i)/r_{gt,c}$. For the calculation of the ratio $r_{gt,c}$ and the training of the survival model, only the ground truth data of the training set is used. Thus, the metrics $r$ and $v$ for instances in the test dataset can be determined without knowledge of ground truth. In summary, we use the following set of metrics $V^i = \{U^i\} \cup \{s, f, d_c, d_s, v, r\}$.

## C. IoU Predictions

The intersection over union is a measure to determine the prediction accuracy of segmentation networks with respect to the ground truth. A predicted instance $i$ is matched with a ground truth instance $g$ if its overlap is the highest compared to the other ground truth instances. The $IoU$ (see eq. (1)) is then calculated between these two instances $i$ and $g$. In this work, we perform instance-wise predictions of the $IoU$

(meta regression) comparing different regression approaches. In addition, we classify between $IoU < 0.5$ and $IoU \geq 0.5$ (meta classification) for all predicted instances. If the $IoU$ of an instance is less than 50%, this instance is considered as a false positive. The metrics introduced in section III-B serve as input for both prediction tasks. As for the survival model, we compute time series of these metrics. We have for an instance $i \in \hat{\mathcal{I}}_{x_t}$ in frame $t$ the metrics $V_t^i$, as well as $V_{t'}^i$ from previous frames $t' < t$ due to object tracking. Meta classification and regression are performed by means of the metrics $V_k^i$, $k = t - n_c, \ldots, t$, where $n_c$ describes the number of considered frames. For regression and classification we use gradient boosting regression (GB) [42], a shallow neural network containing only a single hidden layer with 50 neurons (NN L2) and linear/logistic regression with $\ell_1$-penalization (LR L1). Linear regression with $\ell_1$-penalization is also known as LASSO [43]. Our aim is to analyze how much we benefit from using time series and to which extent the additional metrics $V^i \setminus U^i$ yield improvements.

## IV. Numerical Results

In this section, we evaluate our light-weight tracking algorithm and investigate the properties of the metrics defined in the previous section. We perform meta regression and classification and study the influence of the length of the time series considered as well as different methods for the meta tasks. Furthermore, we study to which extent false positive detection can be traded for additional object detection performance and thus serve as an advanced score. To this end, we compare meta classification with ordinary score thresholds in terms of numbers of false positives and false negatives. We perform our tests on the KITTI dataset [13] for multi-object tracking and instance segmentation, which contains 21 street scene videos from Karlsruhe (Germany) consisting of $1{,}242 \times 375$ 8,008 images. Additionally, we use the MOT dataset [14] with scenes from pedestrian areas and shopping malls, which consists of 2,862 images with resolutions of $1{,}920 \times 1{,}080$ (3 videos) and $640 \times 480$ (1 video). For both datasets annotated videos are available [36]. In contrast to the KITTI dataset, the MOT dataset only includes labels for the class pedestrian, but not for car. In our tests, we consider two different networks, the YOLACT network [7] and the Mask R-CNN [8] with a publicly available implementation [44]. The YOLACT network has a slim architecture designed for a single GPU. We retrain the network using a ResNet-50 [45] backbone and starting from backbone weights for ImageNet [46]. We choose 12 image sequences consisting of 5,027 from the KITTI dataset (same splitting as in [36]) and 300 images from the MOT dataset for training. As validation set we use the remaining 9 sequences of the KITTI dataset, achieving a mean average precision ($mAP$) of 57.06%. The Mask R-CNN focuses on high-quality instance-wise segmentation masks. As backbone we use a ResNet-101 and weights for COCO [47]. We choose the same 12 videos of the KITTI dataset as training set as well as 9 videos as validation set achieving a $mAP$ of 89.87%. In the training of both networks, the validation set

TABLE I
Mismatch ratio and $MOTA$ results obtained by TrackR-CNN and by our tracking approach.

| | TrackR-CNN | | ours | |
|---|---|---|---|---|
| | $\overline{mme}$ | $MOTA$ | $\overline{mme}$ | $MOTA$ |
| MOT | 0.0117 | 0.6657 | 0.0185 | 0.6589 |
| KITTI | 0.0153 | 0.7993 | 0.0235 | 0.7911 |

is neither used for parameter tuning nor early stopping. We choose a score threshold of 0 to use all predicted instances for further experiments. For instance tracking (see algo. 1), we use the following values: $c_o = 0.35$, $c_d = 100$ and $c_l = 50$. In our experiments, we use metrics extracted from the segmentation network's softmax output. The Mask R-CNN outputs a probability distribution per pixel, while the YOLACT network only returns one probability per instance. For this reason, the set $U^i$ consists of more metrics for Mask R-CNN.

### A. *Evaluation of our Tracking Algorithm*

For the evaluation of our tracking algorithm, we use common object tracking metrics, such as the multiple object tracking precision ($MOTP$) and accuracy ($MOTA$) [48]. The $MOTP$ is the averaged distance between geometric centers (geo) or bounding box centers (bb) of matched ground truth and predicted instances. The $MOTA$ is based on three error ratios, the ratio of false negatives, false positives and mismatches ($\overline{mme}$). A mismatch error occurs when the id of a predicted instance that was matched with a ground truth instance changes. In addition, we define by $GT$ all ground truth objects of an image sequence which are identified by different ids and divide these into three cases (see [14]). An object is mostly tracked ($MT$) if it is tracked for at least 80% of frames (out of the total number of frames in which it appears), mostly lost ($ML$) if it is tracked for less than 20%, else partially tracked ($PT$).

In [36] the TrackR-CNN is presented which tracks objects and performs instance segmentation, both by means of a single neural network. This method is tested on the MOT dataset and the validation dataset of KITTI. To compare our approach with the TrackR-CNN method, we use the instances predicted by this network and apply our tracking algorithm to these instances to assess the tracking quality (not the instance prediction quality of the network). The performance metrics mainly evaluate the object detection, while object tracking is only evaluated in the ratio of mismatches and $MOTA$. For this reason, we consider the latter performance metrics for our comparison on the TrackR-CNN instances. The results are given in table I. We obtain for the MOT dataset a slightly higher mismatch ratio $\overline{mme}$ than TrackR-CNN and a $MOTA$ value which is only 0.68 percent points lower. For the KITTI dataset the results are similar. In summary, we are slightly weaker in tracking performance than TrackR-CNN, however, our algorithm does not use deep learning and is independent of the choice of instance segmentation network which enables us to conduct time-dynamic uncertainty quantification for any given instance segmentation network.

TABLE II
OBJECT TRACKING RESULTS AND PERFORMANCE MEASURES FOR THE KITTI DATASET.

| | $MOTP_{bb}$ | $MOTP_{geo}$ | $MOTA$ | $\overline{mme}$ | precision |
|---|---|---|---|---|---|
| YOLACT | 3.34 | 2.68 | $-0.8746$ | 0.0539 | 0.3186 |
| Mask | 2.61 | 2.39 | 0.1281 | 0.0564 | 0.5546 |
| | $GT$ | $MT$ | $PT$ | $ML$ | recall |
| YOLACT | 219 | 124 | 75 | 20 | 0.7211 |
| Mask | 219 | 204 | 13 | 2 | 0.9379 |

TABLE III
CORRELATION COEFFICIENTS $\rho$ WITH RESPECT TO $IoU$.

| | $s$ | $f$ | $d_c$ | $d_s$ | $v$ | $r$ |
|---|---|---|---|---|---|---|
| YOLACT | 0.8760 | 0.8033 | 0.4106 | 0.2755 | 0.8141 | 0.7415 |
| Mask | 0.7303 | 0.6033 | $-0.1082$ | 0.1199 | 0.2049 | $-0.1007$ |

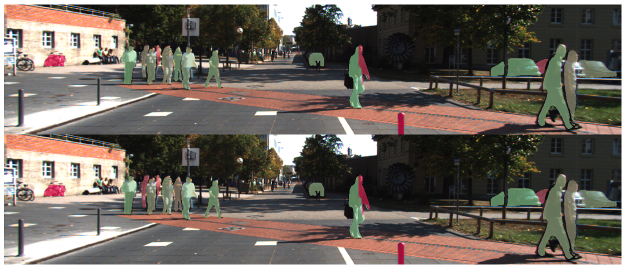
Fig. 2. A visualization of the true instance-wise $IoU$ of prediction and ground truth (*top*) and its prediction obtained from meta regression (*bottom*). Green color corresponds to high $IoU$ values and red color to low ones. Corresponding ground truth and instance prediction are shown in fig. 1.

For further tests, we use YOLACT and Mask R-CNN for instance prediction. The results of object tracking metrics and performance measures for the KITTI dataset are shown in table II. Mask R-CNN consistently achieves better results than YOLACT. This can be attributed to the fact that Mask R-CNN achieves higher $mAP$ values than YOLACT (which is further discussed in section IV-C).

## B. Meta Classification and Regression

In all upcoming experiments, we only use the KITTI dataset. While the KITTI dataset contains 10 frames per second (fps) videos of street scenes, the MOT dataset contains mostly 30 fps videos of pedestrian scenes. Due to slower motions, the data extracted from the images is very redundant and we get fewer different instances. This results in significant overfitting problems for the meta tasks. Also downsampling the frame rate is not an option due to the lack of data. Hence, we only consider the KITTI dataset for further experiments.

For meta classification (false positive detection: $IoU < 0.5$ vs. $IoU \geq 0.5$) and meta regression (prediction of the $IoU$), we use the KITTI validation set consisting of 2,981 images. In our tests, we choose relatively low score thresholds, i.e., 0.1 for YOLACT and 0.4 for Mask R-CNN. In this way, we balance the number of false positives such that the meta tasks obtain enough training data and the tracking performance is not degraded too much. In order to investigate the predictive power of the metrics, we compute the Pearson correlation coefficients $\rho$ between the metrics $V^i \setminus U^i$ and the $IoU$ (see table III). The score value $s$ as well as the shape preservation metric $f$ show a strong correlation with the $IoU$ for both networks. For YOLACT the survival metric $v$ and the ratio $r$ demonstrate high correlations. For meta tasks, we use a combination of metrics and further investigate their predictive power as well as the influence of time series. First, we only present the instance-wise metrics $V^i_t$ of a single frame $t$ to the meta classifier/regressor, secondly we extend the metrics to time series with a length of up to 10 previous time steps $V^i_k$, $k = t-10, \dots, t-1$. For the presented results, we apply a (meta) train/validation/test splitting of 70%/10%/20% and average the results over 10 runs obtained by randomly sampling the splitting. The corresponding standard deviations are given in tables and figures.

For the YOLACT network, we obtain roughly 13,074 instances (not yet matched over time) of which 4,486 have an $IoU < 0.5$. For the Mask R-CNN this ratio is 17,211/6,614. For meta classification, we consider as performance measures the classification accuracy and AUROC. For meta regression, we state the standard errors $\sigma$ and $R^2$ values. The best results (over the course of time series lengths) for meta classification and meta regression are given in table IV. Gradient boosting shows the best performance in comparison to linear models and neural networks with respect to all classification and regression measures. For meta classification, we achieve AUROC values of up to 98.78%. For meta regression, the highest $R^2$ value of 86.85% for the Mask R-CNN network is obtained by incorporating 10 previous frames. For this specific case, an illustration of the resulting quality estimate is shown in fig. 2. We also provide video sequences that visualize the $IoU$ prediction and instance tracking, see https://youtu.be/6SoGmsAarTI. In fig. 3 (left) results for regression $R^2$ as functions of the number of frames are given. We observe that the methods benefit from temporal information, although there are significant differences between them. In fig. 3 (right), we compare our best results for meta classification and regression for both networks with the following baselines. The results in [2] were compared with a single-metric baseline as the mean entropy. We apply as single-metric also the mean entropy per instance as well as the score value using single frame gradient boosting. In addition, the approach in [3] can be considered as a baseline. For this, we only apply the metrics $U^i$ for a single frame and the linear models. For the time-dynamic extension [2] as baseline, the metrics $U^i$ are also used, however as time series and as input for Mask R-CNN and YOLACT. The best results for each method are displayed in this figure. Indeed, our method outperforms all baselines. In fig. 3 (middle) a comparison is shown between the metrics $U^i$, all single frame metrics ($U^i$ plus score and ratio) and all metrics including the temporal ones. As shown before, the metrics $U^i$ are clearly outperformed. The single frame metrics obtain significantly lower $R^2$ values compared to all metrics $V^i$. This difference can be observed when applying linear models as well as neural networks, whereas it is rather small when using gradient boosting. Gradient boosting has a strong tendency to overfitting and due to the small amount

TABLE IV
RESULTS FOR META CLASSIFICATION AND REGRESSION FOR THE DIFFERENT META CLASSIFIERS AND REGRESSORS. THE SUPER SCRIPT DENOTES THE NUMBER OF FRAMES WHERE THE BEST PERFORMANCE AND IN PARTICULAR THE GIVEN VALUES ARE REACHED.

| | YOLACT | | | Mask R-CNN | | |
|---|---|---|---|---|---|---|
| | LR L1 | GB | NN L2 | LR L1 | GB | NN L2 |
| Meta Classification $IoU$ $< 0.5, >= 0.5$ | | | | | | |
| ACC | $90.22\% \pm 3.06\%^2$ | **92.62%** $\pm 2.48\%^9$ | $90.49\% \pm 2.34\%^1$ | $93.43\% \pm 1.78\%^4$ | **95.07%** $\pm 1.24\%^5$ | $94.31\% \pm 1.65\%^4$ |
| AUROC | $95.01\% \pm 1.60\%^6$ | **96.98%** $\pm 0.87\%^9$ | $95.04\% \pm 1.19\%^3$ | $98.26\% \pm 0.86\%^3$ | **98.78%** $\pm 0.53\%^6$ | $98.42\% \pm 0.68\%^6$ |
| Meta Regression $IoU$ | | | | | | |
| $\sigma$ | $0.164 \pm 0.013^7$ | **0.130** $\pm 0.018^5$ | $0.141 \pm 0.021^2$ | $0.170 \pm 0.017^8$ | **0.142** $\pm 0.020^{11}$ | $0.148 \pm 0.025^4$ |
| $R^2$ | $67.34\% \pm 4.10\%^7$ | **79.87%** $\pm 2.66\%^5$ | $75.89\% \pm 5.19\%^2$ | $81.36\% \pm 3.76\%^8$ | **86.85%** $\pm 3.96\%^{11}$ | $85.84\% \pm 3.94\%^4$ |

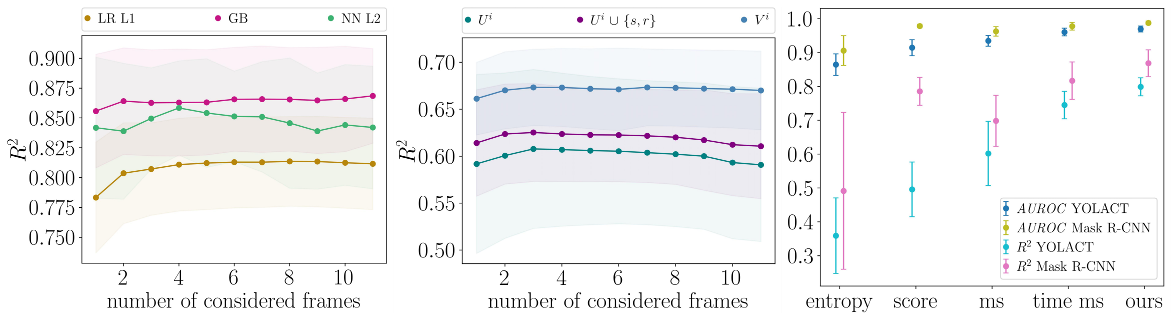

Fig. 3. Results for meta regression $R^2$ as functions of the number of frames for the Mask R-CNN (*left*). Meta regression via linear regression with $\ell_1$-penalization for various input metrics and for the YOLACT network (*middle*). Different baselines for both networks comparing $AUROC$ and $R^2$ values (*right*). From left to right: mean entropy, score value, single-frame MetaSeg (ms) approach [3] with linear models, the time-dynamic extension [2] using metrics $U^i$ and the best performing meta model, our method.

of data, we suspect that the gap between the performance of different metrics would also increase with more data.

In summary, we outperform all baselines by using our time-dynamic metrics as input for meta classifiers/regressors. We reach AUROC values of up to 98.78% for classifying between true and false positives.

## C. Advanced Score Values

In object detection, the score value describes the confidence of the network's prediction. During inference, a score threshold removes false positives. If the threshold is raised to a higher value, not only false positives are removed, but also true positives. We study the network's detection performance while varying the score threshold. In total, we select 30 different score thresholds from 0.01 to 0.98. Meta classification provides a probability of observing a false positive given a predicted instance. We threshold on this probability also with 30 different thresholds and compare this to ordinary score thresholding. To this end, we feed gradient boosting as meta classifier with all metrics $V^i$ including 5 previous frames. In fig. 4 the performance is stated in terms of the number of remaining false positives and false negatives. Each point represent one of the chosen thresholds. For the YOLACT network, the meta classification achieves a lower number of errors, i.e., less false positives and false negatives. In comparison to score thresholding, we can reduce the number of false positives by up to 44.03% for the approximate same number of false negatives. This performance increase is also reflected in the $mAP$. The highest $mAP$ value obtained by score thresholding is 57.04% while meta classification achieves 58.22%. When

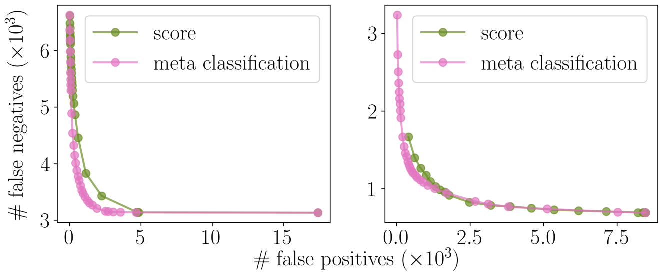

Fig. 4. Number of false positive vs. false negative instances for the YOLACT network (*left*) and Mask R-CNN (*right*) for different thresholds.

applying the Mask R-CNN, we can reduce the number of false positives up to 43.33% for the approximate same number of false negatives. The maximum $mAP$ values for both methods are similar. Score thresholding achieves an $mAP$ of up to 89.87% and meta classification of up to 89.83%. For both networks, we can improve the networks' performance by reducing false positives while the number of false negatives remains almost unchanged. Our method can be applied after training an instance segmentation network and therefore does not increase the network capacity. Up to some adjustment in the considered metrics, this approach is applicable to all instance segmentation networks.

## V. CONCLUSION AND OUTLOOK

In this work, we proposed post processing methods for instance segmentation networks, namely meta classification and regression. These methods are based on temporal information and can be used for both uncertainty quantification and accuracy improvement. We introduced a light-weight tracking algorithm for instances that is independent of the instance segmentation network. From tracked instances we generated time-dynamic metrics and used these as inputs for the meta tasks. In our tests, we studied the influence of our metrics on various time series lengths and different models for the meta tasks. We have shown that our method outperforms baseline methods. Using meta classification we also improved the networks' prediction accuracies by replacing the score threshold with the estimated probability of correct classification during inference.

As an extension of our work, it might be interesting to develop further metrics based on the respective network structure. As an example, consider the YOLACT network where the instance masks are formed by prototypes and associated coefficients.

## References

[1] T. DeVries and G. W. Taylor, "Leveraging uncertainty estimates for predicting segmentation quality," *CoRR*, vol. abs/1807.00502, 2018.

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

[3] M. Rottmann, P. Colling, T. Hack, F. Hüger, P. Schlicht *et al.*, "Prediction error meta classification in semantic segmentation: Detection via aggregated dispersion measures of softmax probabilities," in *International Joint Conference on Neural Networks, IJCNN 2020 Glasgow (UK), July 19-24, 2020*. IEEE, 2020.

[4] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ser. ICML'16. JMLR.org, 2016, pp. 1050–1059.

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

[6] M. T. Le, F. Diehl, T. Brunner, and A. Knol, "Uncertainty estimation for deep neural object detectors in safety-critical applications," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018, pp. 3873–3878.

[7] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, "Yolact: Real-time instance segmentation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 9156–9165.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2980–2988.

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

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

[11] R. Chan, Matthias, Rottmann, F. Hüeger, P. Schlicht, and H. Gottschalk, "Metafusion: Controlled false-negative reduction of minority classes in semantic segmentation," in *International Joint Conference on Neural Networks, IJCNN 2020 Glasgow (UK), July 19-24, 2020*. IEEE, 2020.

[12] P. Jaccard, "The distribution of the flora in the alpine zone," *New Phytologist*, vol. 11, no. 2, pp. 37–50, Feb. 1912.

[13] 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.

[14] A. Milan, L. Leal-Taixé, I. D. Reid, S. Roth, and K. Schindler, "Mot16: A benchmark for multi-object tracking," *ArXiv*, vol. abs/1603.00831, 2016.

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

[16] H. Lee, S. T. Kim, N. Navab, and Y. Ro, "Efficient ensemble model generation for uncertainty estimation with bayesian approximation in segmentation," 05 2020.

[17] K. Wickstrøm, M. Kampffmeyer, and R. Jenssen, "Uncertainty and interpretability in convolutional neural networks for semantic segmentation of colorectal polyps," *CoRR*, vol. abs/1807.10584, 2018.

[18] P.-Y. Huang, W.-T. Hsu, C.-Y. Chiu, T.-F. Wu, and M. Sun, "Efficient uncertainty estimation for semantic segmentation in videos," in *European Conference on Computer Vision (ECCV)*, 2018.

[19] A. G. Roy, S. Conjeti, N. Navab, and C. Wachinger, "Inherent brain segmentation quality control from fully convnet monte carlo sampling," *ArXiv*, vol. abs/1804.07046, 2018.

[20] K. Hoebel, V. Andrearczyk, A. Beers, J. Patel, K. Chang *et al.*, "An exploration of uncertainty information for segmentation quality assessment," 01 2020.

[21] M. Schubert, K. Kahl, and M. Rottmann, "Metadetect: Uncertainty quantification and prediction quality estimates for object detection," 2020.

[22] M. Rottmann and M. Schubert, "Uncertainty measures and prediction quality rating for the semantic segmentation of nested multi resolution street scene images," *CoRR*, vol. abs/1904.04516, 2019.

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

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

[25] D. Miller, L. Nicholson, F. Dayoub, and N. Sünderhauf, "Dropout sampling for robust object detection in open-set conditions," 10 2017.

[26] D. Morrison, A. Milan, and E. Antonakos, "Uncertainty-aware instance segmentation using dropout sampling," 2019.

[27] J. Zhu, H. Yang, N. Liu, M. Kim, W. Zhang *et al.*, "Online multi-object tracking with dual matching attention networks," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[28] J. Son, M. Baek, M. Cho, and B. Han, "Multi-object tracking with quadruplet convolutional neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[29] D. Zhao, H. Fu, L. Xiao, T. Wu, and B. Dai, "Multi-object tracking with correlation filter for autonomous vehicle," *Sensors*, vol. 18, p. 2004, 06 2018.

[30] Z. Wang, L. Zheng, Y. Liu, and S. Wang, "Towards real-time multi-object tracking," *ArXiv*, vol. abs/1909.12605, 2019.

[31] C. Kim, F. Li, and J. M. Rehg, "Multi-object tracking with neural gating using bilinear lstm," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[32] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, and P. H. S. Torr, "Fast online object tracking and segmentation: A unifying approach," *CoRR*, vol. abs/1812.05050, 2018.

[33] C. Aeschliman, J. Park, and A. C. Kak, "A probabilistic framework for joint segmentation and tracking," 07 2010, pp. 1371 – 1378.

[34] C. Payer, D. Stern, T. Neff, H. Bischof, and M. Urschler, "Instance segmentation and tracking with cosine embeddings and recurrent hourglass networks," *CoRR*, vol. abs/1806.02070, 2018.

[35] L. Yang, Y. Fan, and N. Xu, "Video instance segmentation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 5187–5196.

[36] P. Voigtlaender, M. Krause, A. Ošep, J. Luiten, B. B. G. Sekar *et al.*, "MOTS: Multi-object tracking and segmentation," in *CVPR*, 2019.

[37] G. Bertasius and L. Torresani, "Classifying, segmenting, and tracking object instances in video with mask propagation," *ArXiv*, vol. abs/1912.04573, 2019.

[38] J. Luiten, P. Torr, and B. Leibe, "Video instance segmentation 2019: A winning approach for combined detection, segmentation, classification and tracking." in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, 2019, pp. 709–712.

[39] S. Bullinger, C. Bodensteiner, and M. Arens, "Instance flow based online multiple object tracking," 09 2017, pp. 785–789.

[40] D. Moore, *Applied Survival Analysis Using R*, 01 2016.

[41] D. R. Cox, "Regression models and life-tables," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 34, no. 2, pp. 187–220, 1972.

[42] J. H. Friedman, "Stochastic gradient boosting," *Comput. Stat. Data Anal.*, vol. 38, no. 4, pp. 367–378, Feb. 2002.

[43] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society: Series B*, vol. 58, pp. 267–288, 1996.
[44] W. Abdulla, “Mask r-cnn for object detection and instance segmentation on keras and tensorflow,” 2017, github repository.
[45] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
[46] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh *et al.*, “ImageNet Large Scale Visual Recognition Challenge,” *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.
[47] T.-Y. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona *et al.*, “Microsoft coco: Common objects in context,” *ArXiv*, vol. abs/1405.0312, 2014.
[48] K. Bernardin and R. Stiefelhagen, “Evaluating multiple object tracking performance: The clear mot metrics,” *EURASIP Journal on Image and Video Processing*, vol. 2008, 01 2008.