Mitigating Distributional Shift in Semantic Segmentation via Uncertainty Estimation From Unlabeled Data

David S. W. Williams, Daniele De Martini, *Member, IEEE*, Matthew Gadd, *Member, IEEE*, and Paul Newman, *Fellow, IEEE*

***Abstract*—Knowing when a trained segmentation model is encountering data that is different to its training data is important. Understanding and mitigating the effects of this play an important part in their application from a performance and assurance perspective—this being a safety concern in applications such as autonomous vehicles. This article presents a segmentation network that can detect errors caused by challenging test domains without any additional annotation in a single forward pass. As annotation costs limit the diversity of labeled datasets, we use easy-to-obtain, uncurated and unlabeled data to learn to perform uncertainty estimation by selectively enforcing consistency over data augmentation. To this end, a novel segmentation benchmark based on the sense-assess-eXplain (SAX) is used, which includes labeled *test* data spanning three autonomous-driving domains, ranging in appearance from dense urban to off-road. The proposed method, named γ-SSL, consistently outperforms uncertainty estimation and out-of-distribution techniques on this difficult benchmark—by up to 10.7% in area under the receiver operating characteristic curve and 19.2% in area under the precision-recall curve in the most challenging of the three scenarios.**

***Index Terms*—Autonomous vehicle (AV) navigation, deep learning in robotics and automation, introspection, out-of-distribution (OoD) detection, performance assessment, semantic scene understanding, uncertainty estimation.**

I. Introduction

SEMANTIC segmentation is crucial for visual understanding, as semantic information is useful for many robotics tasks, e.g., planning, localization, and mapping [1], [2], [3]. Significant progress has been made on *supervised* semantic segmentation, where accuracy has significantly improved over the years. However, this has mostly involved test datasets from the *same* underlying data distribution as the training data. Considering data with a *distributional shift* from the labeled training data, it is still a significant challenge to train a segmentation network to retain its accuracy on this data and report accurate uncertainty estimates.

The operational design domain (ODD) for a mobile robot is defined as the set of operating conditions under which the robot has been designed to operate safely [5]. However, due to the dynamic nature of uncontrolled outdoor environments, these operating conditions are liable to change, e.g., weather changes, dynamic objects of unknown classes or appearance are seen, illumination varies, etc. This is exacerbated by the significant cost of labeling images for semantic segmentation, as it is intractably difficult to anticipate and represent the full breadth of possible situations in the sample distribution of the labeled training dataset. Therefore, it is crucial that robots are able to verify whether they are in their ODD and can operate safely or whether the domain—and thus, the data distribution—has deleteriously changed. Standards for autonomous vehicles (AVs) [5], [6] cite this as critical for safe deployment.

This work, therefore, answers the question: given a labeled training dataset in one domain (a.k.a. the *source domain*), how can the segmentation error rate be mitigated on a shifted unlabeled domain (a.k.a. the *target domain*)? In answering the question, a model is presented that can learn to perform high-quality uncertainty estimation from an uncurated unlabeled dataset of the target domain, without the prohibitive cost of labeling. An example of the system working is shown in Fig. 1. Here, the vast majority of the image pixels are segmented correctly but the unknown class "horse" is segmented poorly. This is dangerous, e.g., if identified as some other *static* class, the robot may drive forward, or if identified as some other *dynamic* class, downstream prediction or tracking systems may be affected. Both situations mean the robot would act unsafely around a wild animal. Our model however accompanies this prediction with high uncertainty. Critically, this is learned from unlabeled examples in this domain.

This is accomplished by training a segmentation network using a semi-supervised task, where—in lieu of labels—segmentation consistency in the target domain is selectively maximized across data augmentation. The intuition is that the performance on the semi-supervised task can be considered a proxy for segmentation accuracy, which is then used to train the network to express uncertainty on regions of images with poor performance. The proposed network expresses uncertainty in

Manuscript received 31 July 2023; revised 7 January 2024; accepted 27 February 2024. Date of publication 14 May 2024; date of current version 19 June 2024. This paper was recommended for publication by Associate Editor F. T. Pokorny and Editor S. Behnke upon evaluation of the reviewers' comments. This work was supported by EPSRC Programme under Grant EP/M019918/1. *(Corresponding author: David S. W. Williams.)*

The authors are with the Oxford Robotics Institute, Department of Engineering Science, University of Oxford, OX1 2JD Oxford, U.K. (e-mail: dw@robots.ox.ac.uk; danieleg@robots.ox.ac.uk; mattgadd@robots.ox.ac.uk; pnewman@robots.ox.ac.uk).

Digital Object Identifier 10.1109/TRO.2024.3401020

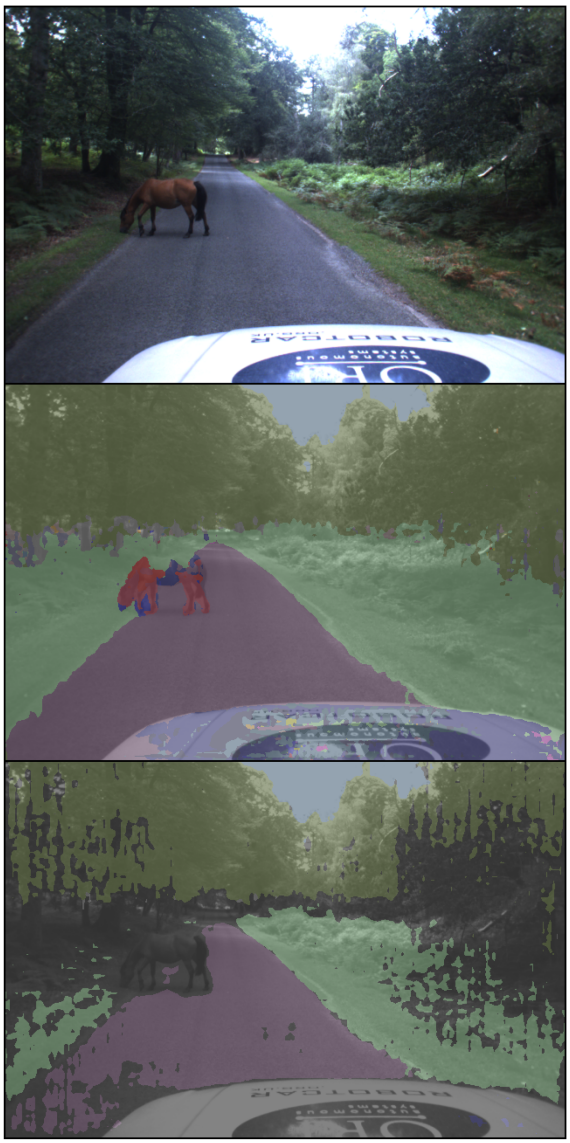


Fig. 1. In an image from the sense-assess-eXplain (SAX) project [4] (top), a horse (an object of unknown class) can be seen on the road. In the central image, this horse is poorly segmented leading to a dangerous driving situation. However, the model proposed in this work expresses pixel-wise uncertainty (blacked pixels on the bottom image), thereby mitigating the poor segmentation and the dangerous situation more generally. Uncertainty is also expressed over unfamiliar greenery that the model struggles to consistently segment as either vegetation or terrain (classes defined in Cityscapes).

feature space with a single forward pass, thus satisfying run-time requirements for a robotics deployment.

The contributions of this article are as follows.

1) It proposes a training method that leverages an unlabeled dataset to learn pixel-wise uncertainty estimation *alongside* segmentation.
2) It evaluates the robustness of the proposed method against several uncertainty estimation and out-of-distribution (OoD) detection techniques.
3) It presents a new semantic segmentation benchmark based on images belonging to the SAX project [4]. This work proposes over 700 pixel-wise labels for a manually curated set of images spanning three domains that are used for testing both semantic segmentation and uncertainty estimation.[1] In addition to the labeled test data, we also propose a set of metrics to evaluate the quality of a model's uncertainty estimates.

II. Preliminaries on Uncertainty Estimation

A given model's error on test data is often described as originating from two different sources: *epistemic* or *aleatoric* uncertainty. The distinction is that error due to epistemic uncertainty is *reducible*, meaning the modeler can reduce the model's test error, e.g., by labeling more data or improving network architecture. In contrast, aleatoric uncertainty is *irreducible* as the uncertainty is inherent in the test data. This means it is impossible to train a model that fully reduces the error on this test data, as aleatoric uncertainty is not under the control of the modeler.

Error due to distributional shift, i.e., distributional uncertainty, has the following two components:

1) uncertainty due to unknown classes, i.e., those not defined in the training data;
2) uncertainty due to known classes with unfamiliar appearance.

Given the constraints of this work, it can be argued that (1) is aleatoric as no model parameterization will fully mitigate the error in the target domain. This is because our model estimates the probability of a pixel belonging to a *fixed* number of defined classes, meaning that class assignment to a novel class is impossible. However, it can be estimated directly from data whether a pixel *does not* belong to a defined class.

Component (2) is caused by the intra-class visual dissimilarity between source and target images. This relates to epistemic uncertainty as it can be reduced by a more diverse labeled training dataset. It can however also be estimated directly from the input target image.

It is, therefore, true that the entirety of distributional uncertainty, i.e., components (1) and (2), can be estimated directly from the data. This motivates this work to draw upon methods for aleatoric uncertainty estimation, as discussed further in Sections III-A and III-C.

III. Related Work

A. Epistemic Uncertainty Estimation

Epistemic uncertainty estimation considers the weight posterior distribution $p(\mathrm{w}|\mathrm{D})$, with w and D defining the model weights and training data, respectively. This distribution over possible model parameterizations given the training dataset is then related to the distribution over possible segmentations for an image. However, this Bayesian analysis is intractable to perform exactly, and so approximations are made, such as Monte Carlo dropout (MCD) [7]. Alternatively, $p(\mathrm{w}|\mathrm{D})$ can be defined using an ensemble of models [8], each trained independently but with the same labeled dataset.

[1]The labels can be found here: https://ori.ox.ac.uk/publications/datasets/.

For training and inference, these methods estimate uncertainty by perturbing the model weights to induce a distribution of segmentations for a given image. For this reason, they are computationally expensive at inference time, as they require multiple forward passes of a network to produce an uncertainty estimate. When considering the deployment of a segmentation network, both latency and memory usage are critical criteria.

This issue is mitigated in [9] by distilling an MCD model into a deterministic network that can estimate uncertainty in a single pass. During the training of our proposed model, a segmentation distribution is instead obtained by applying perturbations to the input data while keeping the model weights constant; at inference time, our model produces segmentation uncertainties in a single pass.

B. Deterministic Uncertainty Methods (DUMs)

Noting the computational requirements of many epistemic uncertainty estimation techniques, DUMs design networks to estimate uncertainty in a single pass using spectral-norm layers [10], which constrain the network's Lipschitz constant, ensuring that semantic differences in the input produce proportionally scaled differences in feature space. Uncertainty can then be estimated as distance in feature space, i.e., the semantic dissimilarity, between a given input and the labeled training data. DUMs differ in how they measure uncertainty in feature space, either with Gaussian processes [11], a post hoc Gaussian mixture model [12], or radial basis function (RBF) kernels [13].

DUMs, like this work, turn uncertainty estimation into a representation learning problem, rather than an study of the model weights. Both methods measure uncertainty in feature space, but DUMs regularize the feature space with layer normalization, whereas this work leverages unlabeled data.

C. Aleatoric Uncertainty Estimation

As aleatoric uncertainty is inherent to the data (rather than the model), aleatoric techniques are designed to estimate uncertainty purely as a function of the input data. This is typically achieved by supervised training. For the training images, the variability in network output with respect to the ground truth is approximated by a distribution [14], [15].

For a regression task in [14], with network estimate $f(x^{(i)})$ and ground truth $y^{(i)}$, the following loss for a pixel i is used to distribute the output as Gaussian:

$$L_{NLL}^{(i)} = \frac{1}{2\sigma(x^{(i)})^2}\|y^{(i)} - f(x^{(i)})\|^2 + \frac{1}{2}\log[\sigma(x^{(i)})]^2.$$

Intuitively, this loss function gives the network two paths to minimize the loss. Its estimate can either be closer to the ground truth or it can mitigate the large squared error $\|y^{(i)} - f(x^{(i)})\|^2$ by expressing a large variance $\sigma(x^{(i)})^2$. In [14], this style of objective is given the name *learned loss attentuation*.

Furthermore, Gast and Roth [16] approximate every layer output as a distribution, using assumed density filtering. Novotny et al. [17] learn features for geometric matching in a self-supervised manner, while expressing uncertainty over poor geometric matches.

This work and [17], unlike many techniques, do not require labels to calculate the learned loss-attenuation objective. This work differs from [17], as it is designed to extract semantics rather than just geometry, meaning labels are required to define the semantic classes. This work also expresses uncertainty as a feature-space distance rather than as a network output.

D. OoD Detection

OoD detection attempts to identify instances that appear distributionally distinct from the labeled training data. The difference with uncertainty estimation is that the focus is on the data, rather than on mitigating model error. One set of techniques train a network in a supervised manner on source data, and then, calculates an OoD score from the network's learned representation. The simplest method calculates the maximum softmax-score [18]. Liang et al. [19] also add adversarial perturbations to the input. In [20], the Mahalanobis distance between the input and training data in a series of feature spaces is combined with logistic regression. In [21], a score that is a function of both the features and the logits is used.

Other methods introduce proxy-training tasks to generate an OoD score. In [22], a classifier learns to classify the transformation applied to the input image, using the max-softmax score as an OoD score. In [23], the task of orientation prediction is utilized, resulting in improved OoD performance.

Our work also introduces a task to learn an OoD score, but uses the task to learn a separable representation of in-distribution and OoD pixels, so the task does not have to be run at inference time. In addition, the aforementioned works train only on in-distribution data, instead of improving robustness via leveraging OoD data. Finally, the proposed model performs pixel-wise, rather than image-wise, OoD detection.

Both [24] and [25] use a curated OoD dataset to explicitly separate in-distribution and OoD data. The authors in [26] and [27] generate OoD datasets containing *near-distribution* images, i.e., those at the edge of the training distribution, with a generative adversarial network and data augmentation, respectively. Near-distribution images are used to achieve more robust OoD detection than clearly OoD images, as separating in-distribution and near-distribution images is much more challenging.

Although unlabeled, a purely OoD dataset can require costly curation. They are also less informative for pixel-wise OoD detection, where in testing, OoD instances are contained in in-distribution scenes (and vice versa). In contrast, our work's target-domain dataset does not require curation and contains near-distribution *and* in-distribution *and* OoD instances within the same image. For these reasons, these data represent the opportunity to learn a more robustly separable representation, at the cost of being more challenging to use.

E. Semi-Supervised Learning

Semi-supervised methods extend supervised methods by including a loss on unlabeled data, and are evaluated on their ability to reduce the test error rate. In contrast, our approach seeks to *detect* test errors. These approaches operate orthogonally, but both prevent unknowingly erroneous predictions.

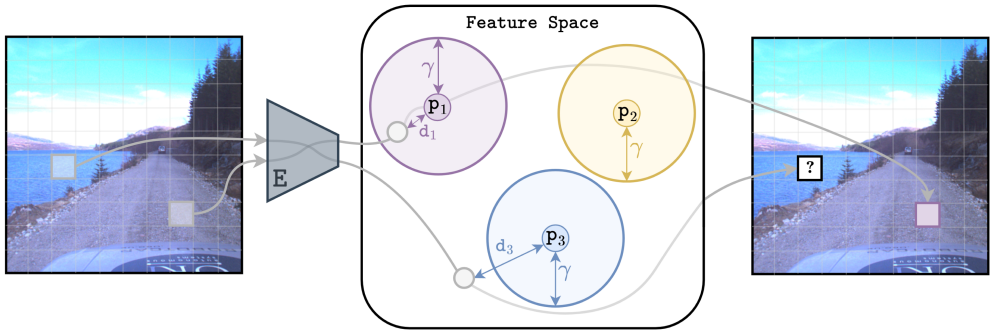


Fig. 2. Depiction of simultaneous segmentation and uncertainty estimation for the model presented in this work. Pixel-wise features are extracted from an image by encoder E. Distances $\mathtt{d}_{1:3}$ are calculated between each feature and prototypical features from each class $\mathtt{p}_{1:3}$, known as prototypes. If one of $[\mathtt{d}_1, \mathtt{d}_2, \mathtt{d}_3] < \gamma$, the feature is certain and assigned the class of its closest prototype (denoted by the colored pixel overlaid on the right), and if not, the feature is assigned uncertain (denoted as the question mark in white pixel). In this way, a "safe region of operation" is defined in feature space, where inside pixels are accurate and certain, and outside they are uncertain and inaccurate.

Semi-supervised approaches often maximize the consistency of a model's representation across perturbations to the input, model, or both [28], [29], [30]. Similar to this work, Assran et al. [31] use data augmentation and a cross-entropy objective between the class assignment distributions. In contrast, we apply the objective selectively and per-pixel. To appropriately represent OoD data, we also introduce additional objectives and procedures in place of the regularization objective seen in [31].

1) Prototypes: Related to [31], Caron et al. [32] maximize consistency using prototypes as part of its mechanism, also seen in few-shot learning literature [33], [34]. A prototype is calculated for each class as the centroid of all source embeddings for that class. Prototypes thus compactly represent the high-level semantics for a given class by averaging over the intraclass factors of variation. In addition, DUMs calculate class mean features (i.e., prototypes) and measure uncertainty as a distance between features and class centres, as mentioned in Section III-B.

2) Unsupervised Domain Adaptation (UDA): A specific instance of semi-supervised learning is UDA, which specifically targets the distributional shift between the labeled (a.k.a. *source*) and unlabeled (a.k.a. *target*) data. Still, UDA methods [35], [36] are designed purely to increase test accuracy by learning an improved representation of the target domain and not to detect the errors arising from the shift, as in our method.

IV. System Overview

Fig. 2 shows an overview of the proposed system. At inference, its goal is to segment each pixel of an image of size $H \times W$, $x \in \mathbb{R}^{3\times W\times H}$ into K known classes $\mathcal{K} = \{k_1, \dots k_K\}$ or flag them as uncertain. This is done by producing both a categorical distribution $p(y|x) = [p(y = k_1|x), \dots p(y = k_K|x)] \in \mathbb{R}^{K\times W\times H}$ and an uncertainty mask $M_\gamma \in \mathbb{B}^{W\times H}$, which assigns to each pixel 1 for certain or 0 for uncertain through a threshold γ in feature space.

A. Segmentation Using Prototypes

An encoder, $\mathtt{E} : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^{F\times h\times w}$, and projection network $\mathtt{g}_\rho : \mathbb{R}^{F\times h\times w} \to \mathbb{R}^{F\times h\times w}$ calculate embeddings $z \in \mathbb{R}^{F\times h\times w}$, $\|z\|_2 = 1$ of an image x, where F is the feature length, and h and w are the downsampled spatial dimensions. Let $X_S \in \mathbb{R}^{N\times 3\times W\times H}$ be a batch of N source images and $Y_S \in \mathbb{R}^{N\times h\times w\times K}$ their corresponding one-hot labels downsampled to the size of the embeddings $Z_S \in \mathbb{R}^{N\times h\times w\times F}$. Prototypes $p_S \in \mathbb{R}^{F\times K}$ are then calculated for each of the K classes from the embeddings Z_S as follows:

$$p_S = \frac{Z_S^\top Y_S}{\|Z_S^\top Y_S\|_2} \tag{1}$$

where $Z_S^\top Y_S \in \mathbb{R}^{F\times w\times h\times N} \times \mathbb{R}^{N\times h\times w\times K} = \mathbb{R}^{F\times K}$, giving us a normalized, aggregate feature per class. A segmentation $p(y|x)$ is obtained by projecting each pixel embedding of x onto the prototypes. Specifically, the classification scores, $\tilde{s}^{(i)} \in \mathbb{R}^{1\times K}$, for the ith pixel embedding $z^{(i)} \in \mathbb{R}^{F\times 1}$ are

$$\tilde{s}^{(i)} = z^{(i)\top} p_S \tag{2}$$

where $z^{(i)\top} p_S \in \mathbb{R}^{1\times F} \times \mathbb{R}^{F\times K} = \mathbb{R}^{1\times K}$. For this segmentation method, the scores $\tilde{s}$ represent the cosine similarity between a feature and the prototype for each class. These scores over the downsampled spatial resolution $\tilde{s} \in \mathbb{R}^{h\times w\times K}$ are then bilinearly upsampled to $s \in \mathbb{R}^{H\times W\times K}$. The categorical distribution over the K classes, $p(y|x)^{(i)} \in \mathbb{R}^K$, is given by

$$p(y|x)^{(i)} = \sigma_\tau(s^{(i)}) \tag{3}$$

where $\sigma_\tau(.)$ is the softmax function with temperature τ (in this work, $\tau = 0.07$).

B. Uncertainty Estimation Using γ

Uncertainty is expressed in this work as the probability that a pixel belongs to *no known class*, i.e., $p(y \notin \mathcal{K}|x)$. Taking inspiration from [37] and [38], we append a parameter γ to the classification scores as the $(K+1)$th score

$$p(y|x)^{(i)} = \sigma_\tau(s^{(i)} \oplus \gamma) \in \mathbb{R}^{K+1} \tag{4}$$

where $\oplus$ denotes vector concatenation.

The largest score, $\max(s^{(i)})$, is the cosine similarity between an embedded pixel and its closest prototype. This is a measure of

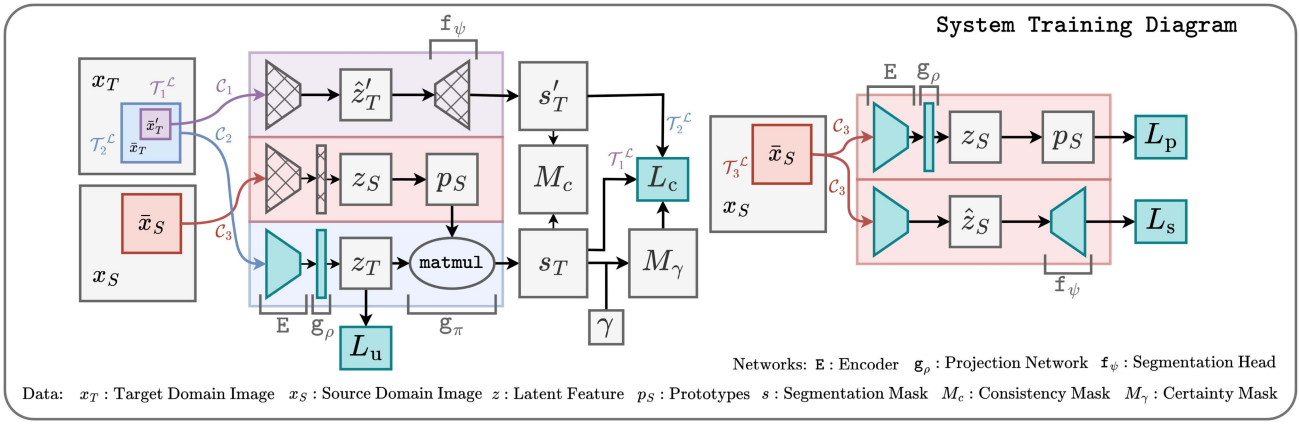


Fig. 3. Training regime of the proposed approach. The model parameters are updated by four losses: (a) L_{c}, (b) L_{u}, (c) L_{p}, and (d) L_{s}. (a) For the pixels deemed certain by M_γ, L_{c} maximizes the consistency—a proxy for accuracy—over the segmentations s'_T, s_T of augmented versions $\bar{x}'_T$, $\bar{x}_T$ of the original target domain image x_T. (b) L_{u} softly constrains the features z_T to be uniformly distributed on the unit-hypersphere. (c) L_{p} maximizes the distance between source prototypes p_S, i.e., spreads the mean embeddings of each class in the source domain dataset over the unit-sphere uniformly. (d) L_{s} maximizes the accuracy for the segmentations of the source images x_S with respect to ground-truth labels. For each diagram, the networks colored in aquamarine are updated by the losses, while the cross-hatched networks are not. Note that for diagrammatic clarity, the color transforms are depicted as following $\bar{x}'_T, \bar{x}_T, \bar{x}_S$, whereas in reality and as described in Section V-A: $\bar{x}'_T = \mathcal{C}_1 \circ \mathcal{T}_1^{\mathcal{L}} \circ \mathcal{T}^{\mathcal{G}}(x_T)$, $\bar{x}_T = \mathcal{C}_2 \circ \mathcal{T}_2^{\mathcal{L}} \circ \mathcal{T}_1^{\mathcal{G}}(x_T)$, $\bar{x}_S = \mathcal{C}_3 \circ \mathcal{T}_3^{\mathcal{L}} \circ \mathcal{T}_2^{\mathcal{G}}(x_S)$. Best viewed in color.

the model's confidence, upon which γ operates as a threshold.[2] Certainty mask, M_γ, is given by

$$M_\gamma^{(i)} = 1 - \mathbb{1}[\mathrm{argmax}(s^{(i)} \oplus \gamma) = K{+}1] \tag{5}$$

where $\mathbb{1}[\cdot]$ is the indicator function, and so certainty is given by $M_\gamma^{(i)} = 1$ if the highest score is a known class, otherwise $M_\gamma^{(i)} = 0$ if γ is the highest score. This way, γ defines a region around each prototype, outside of which the feature is considered uncertain.

V. Training Objectives

Ultimately, we aim to train a model that can express uncertainty with uncertainty map M_γ via the threshold γ. Yet, in the absence of labels, we draw on the hypothesis that segmentation consistency over image augmentation approximates ground-truth accuracy. Consider two corresponding pixels from two augmentations: if both are assigned the same class, this is likely to be accurate; otherwise, it is not. Embedding consistency, whether feature distance [39], [40] or implicit class assignment [32], is a good proxy for classification accuracy, shown by the success of linear probe experiments in training classifiers from models trained with an embedding consistency objective. Instead, we focus on learning a representation for uncertainty estimation, and use pixel-wise alignment, rather than image-wise supervision.

As seen in Fig. 3, consistency between two augmented target-domain images $\bar{x}'_T$ and $\bar{x}_T$, is represented by the consistency map $M_c \in \mathbb{B}^{H \times W}$, where each pixel is 1 if consistent or 0 otherwise.

[2]For example, if $s^{(i)}_{k=2}$ is larger than γ, after a low temperature softmax, $p(y=2|x)^{(i)}$ will be high. However, if none of $s^{(i)}$ surpass γ, then $p(y = K{+}1|x) \equiv p(y \notin \mathcal{K}|x)$ will be high, (see Fig. 2 for a depiction).

In this work, consistency maps are related to uncertainty maps in the following two-step training process.

1) γ is solved for such that the mean certainty of segmentations according to M_γ is equal to the mean consistency according to M_c. γ, therefore, separates pixels into certain and uncertain broadly according to consistency.
2) The model parameters are then updated by maximising the consistency of pixels deemed certain by M_γ, as a proxy for maximising the accuracy of the certain pixels. This both improves the estimate of M_{c} and separates features of pixels assigned certain and uncertain.

This method, therefore, establishes a *positive feedback loop*, where M_c and M_γ continually improve each other's estimates of which pixels are segmented accurately. The following sections describe how, with care, the training dynamics can be conditioned such that this leads to simultaneous high-quality uncertainty estimation and segmentation.

A. Semi-Supervised Task

Augmentations transform one image into two with distinct appearances but contain the same underlying semantics. First, a target image x_T is randomly cropped with transform $\mathcal{T}_1^{\mathcal{G}}$. This global crop is transformed by $\mathcal{T}_1^{\mathcal{L}}$ and $\mathcal{T}_2^{\mathcal{L}}$, which are sampled such that one is always a local crop and resize, and the other an identity transform. Finally, $\bar{x}'_T$ and $\bar{x}_T$ are obtained by applying color-space transforms $\mathcal{C}_1$ and $\mathcal{C}_2$. At the end, $\bar{x}'_T$ and $\bar{x}_T$ are images of the same spatial dimensions, but of different appearance and where one is an upsampled crop of a region within the other.

Both of these images are then segmented by functions $\mathtt{f}$ and $\mathtt{g}$. Pixel-wise segmentations s'_T and s_T are obtained by applying the opposite local cropping transform to the one applied to the input image, as follows:

$$s'_T = \mathcal{T}_2^{\mathcal{L}} \circ \mathtt{f} \circ \mathcal{C}_1 \circ \mathcal{T}_1^{\mathcal{L}} \circ \mathcal{T}_1^{\mathcal{G}}(x_T)$$

Algorithm 1: Algorithm to Calculate γ.

1: **Inputs:**
2: Consistency mask: $M_c \in \mathbb{R}^{N\times H\times W}$
3: Classification scores: $S_T \in \mathbb{R}^{N\times K\times H\times W}$
4: **function** `calculate_gamma`$(M_c), (S_T)$
5: MaxS_T = `flatten(max(`S_T`, dim = "K"))`
6: MaxS_T = `sort(`MaxS_T`, ascending=True)`
7: p_c = `mean(`M_c`)` ▷% consistent pixels
8: R $= (1 - p_c) * N * H * W$ ▷Num. uncertain pixels
9: γ = MaxS_T `[int(R)]`
10: **return** γ
11: **end function**

$$s_T = \mathcal{T}_1^{\mathcal{L}} \circ \mathtt{g} \circ \mathcal{C}_2 \circ \mathcal{T}_2^{\mathcal{L}} \circ \mathcal{T}_1^{\mathcal{G}}(x_T). \tag{6}$$

Here, s'_T and s_T are, therefore, pixel-wise aligned as they are both segmentations of the region of the smaller local crop. $\mathtt{f}(\cdot)$ and $\mathtt{g}(\cdot)$ represent the top and bottom branches in Fig. 3 and are distinct functions that both return a segmentation for a given image. Insights for why $\mathtt{f} \neq \mathtt{g}$ are given in Section V-E.

Transformations are applied to x_S in the following order to obtain $\bar{x}_S$: $\mathcal{T}_2^{\mathcal{G}}, \mathcal{T}_3^{\mathcal{L}}, \mathcal{C}_3$, where $\mathcal{T}_2^{\mathcal{G}}$ is a global crop, $\mathcal{T}_3^{\mathcal{L}}$ is the identity transform or a local crop and resize, and $\mathcal{C}_3$ is a color-space transform.

B. Calculating γ: Making Inconsistent Pixels Uncertain

Let us consider batches of target domain images, $\bar{X}'_T, \bar{X}_T \in \mathbb{R}^{N\times 3\times H\times W}$, and segmenting them to obtain $S'_T, S_T \in \mathbb{R}^{N\times K\times H\times W}$. The consistency mask $M_c \in \mathbb{B}^{N\times H\times W}$, where $M_c^{(i)} = 1$ if consistent, else $M_c^{(i)} = 0$, is given by

$$M_c^{(i)} = \begin{cases} 1 & \underset{k\in\mathcal{K}}{\text{argmax}}(S_T^{'(i)}) = \underset{k\in\mathcal{K}}{\text{argmax}}(S_T^{(i)}) \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where γ is then calculated such that the $\mathrm{p}(\text{certain})$ (i.e., the proportion of pixels that are certain according to M_γ) is equal to $\mathrm{p}(\text{consistent})$ (i.e., the proportion of pixels consistent according to M_c), as detailed in Algorithm 1. Here, MaxS_T contains the largest similarity with prototypes for each pixel. $(1 - p_c)$ is the proportion of inconsistent pixels in the batch. Line 9 then chooses γ so that certain pixels have the same proportion as consistent.

C. Learning E: Making Certain Pixels Consistent

Similar to learned loss attenuation, the ultimate objective is to train a model that produces either high-quality segmentations or expresses high uncertainty. To this end, a consistency objective, L_c, maximizes the quality of the segmentation, but *only* for pixels that are deemed certain by M_γ. Segmentation quality is represented by the consistency, which in this case is calculated as the cross entropy between the pixel-wise categorical distributions across views, as follows:

$$L_\mathrm{c} = \frac{\sum_i^{NHW} M_\gamma^{(i)}\, \mathcal{H}[p(y|\bar{x}'_T)^{(i)}, p(y|\bar{x}_T)^{(i)}]}{\sum_j^{NHW} M_\gamma^{(j)}} \tag{8}$$

where $\mathcal{H}$ is the cross-entropy function. As shown in Fig. 3, only the encoder $\mathtt{E}$ is updated by this loss function.

L_c causes an entropy decrease for $p(y|x)$ of certain pixels, but not for uncertain pixels. Given that $p(y|\bar{x}_T)$ is produced via prototype segmentation (see Section IV-A), this relates to an increase in the separation between the features z_T of certain and uncertain pixels. This is because certain features, i.e., those with a cosine distance less than γ to a prototype (see Fig. 2), have been pulled closer to their closest prototype, and thus further from the uncertain features.

D. Additional Objective Functions

L_c does *not* maximize consistency for uncertain pixels. Despite this, when only L_c is used to update the model, *all* features in the target domain tend to collapse onto a subset of the source prototypes, thereby achieving near-perfect consistency irrespective of the input image.

This negatively affects uncertainty estimation, as the calculated γ cannot effectively separate certain and uncertain features, and so M_γ is a poor uncertainty estimate. In addition, few of the prototypes onto which the features collapse correspond to the correct ground-truth class, so the near-perfect consistency does not correspond to near-perfect accuracy. This results both in an inaccurate model and an M_c that approximates accuracy very poorly.

The proposed solution to this problematic training dynamic is to softly constrain the model to distribute each batch of target features uniformly on the unit hypersphere, as presented in [41]. Uniformity prevents feature collapse by constraining the proportion of features near the prototypes, thus making the model more selective about which features are certain or uncertain. L_u is calculated in the same form as in [41] as follows:

$$L_\mathrm{u} = \frac{1}{N h_\mathrm{u} w_\mathrm{u}} \sum_{i\neq j} e^{-t||\tilde{Z}_T^{(i)} - \tilde{Z}_T^{(j)}||_2^2} \tag{9}$$

where $t = 2$ and $\tilde{Z}_T \in \mathbb{R}^{N\times F\times h_\mathrm{u}\times w_\mathrm{u}}$ is a batch of target features, which has been downsampled by average pooling such that $h_\mathrm{u}, w_\mathrm{u} = h/4, w/4$. Average pooling reduces the number of pairwise distances calculated, reducing memory usage.

Simultaneously, another loss term, L_p [42], maximizes the distance between the source prototypes, i.e., spreads them on the unit sphere, preventing L_u from concentrating the prototypes to maximize the distance between certain and uncertain features. For K class prototypes $p_S \in \mathbb{R}^{F\times K}$

$$L_\mathrm{p} = \frac{1}{K} \sum_{i=1}^{K} \max_{j\in K} [p_S^\top p_S - 2\mathrm{I}]_{ij}. \tag{10}$$

This minimizes the similarity between nearest prototypes, as $p_S^\top p_S \in \mathbb{R}^{K\times K}$ contains cosine similarities between prototypes, and subtracting 2I excludes self-similarity terms.

While L_u and L_p both maximize the distance between features, L_u applies this locally using an RBF kernel, whereas L_p maximizes nearest-neighbor feature distance, and thus, more strongly encourages uniformity.

Finally, a supervised loss is calculated using the source labels to maintain a good representation of the source domain. This objective, L_{s}, is used to update the encoder and segmentation head and is calculated as the cross entropy

$$L_{\text{s}}^{(i)} = -\sum_{k\in\mathcal{K}} y_S^{(i)} \log(p(y=k|x_S)^{(i)}) \tag{11}$$

where $y_S^{(i)}$ is the ground-truth one-hot label for the pixel i.

E. Asymmetric Branches

As seen in Fig. 3, the branches segmenting $\bar{x}'_T$ and $\bar{x}_T$ are not identical. The top branch, $\mathtt{f} : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^{K\times H\times W}$, segments $\bar{x}'_T$ through $\mathtt{E}$ and a segmentation head rather than prototype segmentation; therefore $\mathtt{f}(\cdot) = \mathtt{f}_\psi \circ \mathtt{E}(\cdot)$, where $\mathtt{f}_\psi(\cdot)$ is the segmentation head. In contrast, the bottom branch, $\mathtt{g} : \mathbb{R}^{3\times H\times W} \to \mathbb{R}^{K\times H\times W}$, segments $\bar{x}_T$ as $\mathtt{g}(\cdot) = \mathtt{g}_\pi \circ \mathtt{g}_\rho \circ \mathtt{E}(\cdot)$, where $\mathtt{g}_\rho(\cdot)$ is a projection network and $\mathtt{g}_\pi(\cdot)$ performs prototype segmentation as $\mathtt{g}_\pi(z_T) = z_T^\top p_S$.

Branch asymmetry prevents training from collapsing. L_{c} can be trivially minimized by assigning large regions to the same incorrect class across views, similar to what is described in Section V-D. Similar architectures are found in self-supervised learning methods such as in [40] and [43], also prevent similar failure modes—using exponential moving averages of model weights [40] and using additional layers [43].

As a result, the proposed asymmetry is not susceptible to collapse. Suppose that the encoder $\mathtt{E}$ collapses, for L_{c} to be minimized by $s'_T = s_T$, the following has to be true: $\mathtt{f}_\psi(\cdot) = \mathtt{g}_\pi \circ \mathtt{g}_\rho(\cdot)$. This is not observed, which we attribute to the following factors:

1) $\mathtt{g}_\pi \circ \mathtt{g}_\rho$ and $\mathtt{f}_\psi$ are architecturally different;
2) neither the segmentation head $\mathtt{f}_\psi$, nor the prototypes via $\mathtt{g}_\pi$ can contribute to a degenerate solution, as neither are updated by L_{c};
3) $\mathtt{g}_\rho$ is updated with L_{u} in addition to L_{c}.

A secondary benefit of using the segmentation head $\mathtt{f}_\psi$ is that it naturally produces a low entropy $p(y|\bar{x}'_T)$. This contributes to decreasing the entropy of $p(y|\bar{x}_T)$, thus further separating certain and uncertain pixels in the manner described in Section V-C.

VI. Experimental Setup

This work introduces a novel test benchmark, building on top of the SAX project [4]. The benchmark is composed of pixel-wise labels that annotate a set of manually curated images from three domains of the SAX project. Alongside the test labels, this benchmark also proposes test metrics to evaluate quality of uncertainty estimation, presented in Section VII.

A. Data

This work uses three different types of data: 1) labeled training images; 2) unlabeled training images; and 3) labeled test images. The primary experiments in this work use Cityscapes [44] as 1), and a SAX domain provides 2) and 3). As such, unless otherwise stated, the labeled dataset used is Cityscapes. However, in order to investigate the generality of the method, Berkeley DeepDrive 10 k (BDD) [45] is also considered as a source domain, and both BDD and KITTI [46] are used as target domains .

The SAX dataset comprises data from three domains defined by their location of collection: London, the New Forest (a rural region in southern England), and the Scottish Highlands, ordered by descending similarity with Cityscapes. Examples from each dataset can be seen in Fig. 4. By testing across all three domains, the effect of the magnitude of the distributional shift on uncertainty estimation can be evaluated.

Each domain contains instances that are in-distribution (e.g., cars, road, and signs that look very similar to those found in Cityscapes) and OoD (e.g., classes not defined in Cityscapes such as horses, Scottish lochs, and gravel roads). Classes undefined in source domain are treated such that any assignment to these classes is treated as inaccurate. Importantly, each domain also contains many instances on the edge of the labeled distribution, i.e., *near-distribution*. Each of these instance types are combined within images, causing a significant challenge to uncertainty estimation and OoD detection models.

For the KITTI labeled test dataset, the 201 labeled training images were used (as only these have downloadable labels), while the unlabeled training dataset come from the published raw data. BDD publishes both a labeled training dataset, test dataset, and 100 000 driving images without semantic annotation. This allows us to use BDD as both source and target domain separately. For both KITTI and BDD, care was taken to prevent any overlap between labeled testing images and the unlabeled training images.

B. Network Architecture and Training

For every experiment, the segmentation network used has a DeepLabV3+ architecture [47] with a ResNet18 backbone [48]. More specifically, $\mathtt{E}$ is represented by both the ResNet18 and the ASPP module, and so the features $\hat{z}$ are taken from the penultimate layer of DeepLabV3+, with just the segmentation head $\mathtt{f}_\psi$ to follow.[3] For prototype segmentation, features $\hat{z}$ are then passed through a projection network $\mathtt{g}_\rho$, which is a two-hidden-layer perceptron—similar to [39], but applied to each pixel embedding independently. The feature dimensions are given by $h = {}^H/_4$, $w = {}^W/_4$, $F = 256$.

Before being updated by L_{c}, the networks $\mathtt{E}$ and $\mathtt{g}_\rho$ are pretrained using only L_{s} and L_{u}. First, this means that before semi-supervised training begins, the segmentation head $\mathtt{g}_\rho$ has already broadly learned the spatial distribution of classes. Second, after a small number training iterations with L_{c} and L_{p}, the prototypes faithfully represent the semantic classes due to the pretraining of $\mathtt{E}$. In combination, these factors mean that M_c starts as a better estimate of ground-truth segmentation accuracy, and thus, the system is well-initialized for the positive feedback loop described in Section V.

[3] see https://github.com/qubvel/segmentation_models.pytorch for implementation details

(a) (b) (c) (d)

Fig. 4. Example images and relative segmentation masks from Cityscapes—the source domain—and the domains in the *SAX segmentation test dataset*. (a) Cityscapes. (b) SAX London. (c) SAX New Forest. (d) SAX Scotland.

C. Use of a Domain-Based Curriculum

Models trained with the method presented in Section IV are given the name γ-SSL. For each domain, a separate γ-SSL model is trained, such that testing occurs in the domain of the unlabeled training data. Models named γ-SSL$_{\text{iL}}$ are, however, also initialized on weights trained with unlabeled SAX London images.

γ-SSL$_{\text{iL}}$ models are trained under the following hypothesis: splitting training into two chunks (source $\rightarrow$ intermediate target, intermediate target $\rightarrow$ final target), reduces the distributional gap between source and target for each chunk of training, and this improves the quality of the learned representation of the final target domain. SAX London is this intermediate domain, as it is most similar Cityscapes, while also sharing platform configurations with other SAX domains.

The motivation for this comes from curriculum learning [49], where a model's performance is improved by using a training procedure in which the difficulty of training examples increases during training. In our case, the difficulty of the curriculum is controlled by one high-level characteristic of the domain, i.e., its geographic location, but other characteristics could be used, such as rainy/dry, day/night, and sunny/overcast.

Given that robots are designed for a specific ODD, the source domain is precisely defined by its operating conditions. By considering what the ODD considers in-distribution, the curriculum can be designed to include progressively more OoD conditions. The added diversity in this curriculum naturally increases the data requirements for γ-SSL$_{\text{iL}}$. However, in a robotics context, this work argues that these requirements are not difficult to satisfy. This is because collecting an uncurated unlabeled dataset for a specific set of operating conditions (e.g., those not contained in the ODD) merely requires access to a robot and for those conditions to exist in the real world. This argument also explains why the standard γ-SSL models are not significantly more difficult to train than the benchmarks.

D. Benchmarks

This work evaluates a range of techniques, each producing a likelihood per-pixel of the model making an error. As the test data are distributionally shifted from the labeled training data, the proposed method is benchmarked against several epistemic uncertainty estimation and OoD detection techniques.

Methods are split into *epistemic*- and *representation*-based methods. The distinction is that the epistemic methods consider a distribution over the model parameters, sample from that distribution, and then, calculate the inconsistency in segmentation; this process requires multiple forward passes of the network. Instead, the representation methods solely leverage a learned representation and compute a metric to determine the uncertainty in a single forward pass, greatly reducing the computational requirements for deployment.

The epistemic methods comprise MCD [7] and deep ensemble (Ens) [8]. For both, predictive entropy (PE) and mutual information (MI) are used as uncertainty measures, where MI is more often used to estimate epistemic uncertainty, as evidenced by its use in active learning [50]. The network for MCD builds on Bayesian DeepLab [51], but is adapted for a ResNet18, and is tested over a range of dropout probabilities and numbers of samples, with the best presented (0.2 and 8, respectively). Different ensemble sizes are also evaluated. The MCD model is also distilled into a deterministic network, named MCD-DSL, as per [9].

As for the representation methods, the techniques investigated include several OoD detection methods [18], [19], [20], [21]. Softmax [18] and $\text{Softmax}_{\text{A}}$ [19] propose a pretrained segmentation network with tuned softmax temperature parameters, where the latter also leverages adversarially perturbed images . Lee et al. [20] leverage a pretrained network where OoD score is the Mahalanobis distance in feature space (FeatDist), which can also leverage adversarial inputs ($\text{FeatDist}_{\text{A}}$). Finally, ViM [21] defines the OoD score as a function of both the features and

TABLE I
CONFUSION MATRIX FOR MISCLASSIFICATION DETECTION

Actual \ Predicted	$[\text{certain}]_t$	$[\text{uncertain}]_t$
accurate	TP	FN
inaccurate	FP	TN

The problem is set as a binary-classification with: true positive (TP), false negative (FN), false positive (FP) and true negative (TN). Each of the states certain, uncertain are determined by a given uncertainty threshold t.

the logits. The feature space chosen for [20] and [21] is the same as used for the proposed method. When using adversarially perturbed inputs, evaluation takes place over a range of step-sizes, ϵ. The final representation method is a DUM, presented in [12], and uses the official implementation of [52].

VII. EVALUATION METRICS

This work evaluates methods on their ability to perform misclassification detection. This is a binary classification problem, whereby pixels segmented accurately with respect to labels should be classified as certain, and inaccurate pixels classified as uncertain. These states are defined in Table I.

Given a set of imperfect models, the best model for a binary classification problem can be selected based on a number of different metrics. For uncertainty estimation, the most appropriate metric is dependent on the context in which the uncertainty estimates and segmentation predictions are used. For this reason, this work considers a range of possible definitions of metrics and justifies each in a robotics context.

A. Metrics: Definitions

First, this work considers receiver operating characteristic (ROC) curves and precision-recall (PR) curves for the evaluation of misclassification detection, based on the prior use of these metrics in [18]. ROC curves plot the true positive rate (TPR) versus the false positive rate (FPR) as follows:

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

where TPR is the proportion of accurate pixels detected as certain, whereas FPR is the proportion of inaccurate pixels incorrectly assigned to certain. The ROC curve treats the positive and negative classes separately and is thus independent of the class distribution, i.e., the underlying proportion of pixels segmented as the correct semantic class. The ROC curve is summarized by the area under it, the AUROC. Precision and recall are defined, respectively, as

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}.$$

Interpreting misclassification detection as an information retrieval task, PR curves evaluate the ability to use uncertainty estimates to retrieve *only* accurate pixels (as the positive class is defined as accurate in Table I). As for ROC curves, PR curves can be summarised by the area under them, the AUPR.

In addition, as an alternative to AUPR, the F_β score is also considered, defined as

$$\text{F}_\beta = \frac{(1+\beta^2)\text{TP}}{(1+\beta^2)\text{TP} + \text{FP} + \beta^2\text{FN}}.$$

This factors precision and recall into a single metric, weighting their contribution through the scalar β. For $\beta < 1$ a stronger focus is given to Precision, while for $\beta > 1$ to Recall.

Finally, as in [51], the misclassification detection accuracy, A_MD, is also used to evaluate uncertainty estimation. This is defined for a given uncertainty threshold as

$$\text{A}_\text{MD} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}.$$

According to this metric, the best model is the one which segments the highest proportion of *all pixels* in one of two states: $(\text{accurate}, \text{certain})$ or $(\text{inaccurate}, \text{uncertain})$.

A_MD and $\text{F}_{0.5}$ are plotted against the proportion of pixels that are in the state: $(\text{accurate}, \text{certain})$, named $\text{p}(\text{a}, \text{c})$, where

$$\text{p}(\text{a}, \text{c}) = \frac{\text{TP}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}.$$

For these plots, the ideal model should maximize both the uncertainty metric (A_MD, $\text{F}_{0.5}$) and $\text{p}(\text{a}, \text{c})$, i.e., better results are closer to the top-right of the plots. Therefore, the maximum value of A_MD and $\text{F}_{0.5}$ —named MaxA_MD and $\text{MaxF}_{0.5}$, respectively—and the value of $\text{p}(\text{a}, \text{c})$ at which they occur are also reported.

B. Discussion: AUROC, AUPR, and F-Scores

Given the context of this work, i.e., semantic segmentations for robotics applications, we must consider the following:

1) what the real-world costs are for misclassification in the cases of FP versus FN;
2) whether the evaluation should be independent of the class distribution, $\text{p}(\text{accurate})$

The first is context-dependent to a large extent. In general, for safety-critical contexts such as robotics, where perception directly leads to decisions and actions in the real world, the importance of certain being accurate is higher than uncertain being inaccurate. To simplify, *accidents arise when autonomous systems make confident but incorrect predictions about their surroundings*. In contrast, when predictions are uncertain but accurate, the system is overly conservative, and thus, does not take action as it considers some safe actions unsafe; this is inefficient but less hazardous. It can, therefore, be argued that precision is more important than recall—we want $\text{FP} = 0$, i.e., no pixels inaccurate and certain even if some pixels are accurate but uncertain ($\text{FN} > 0$).

PR and ROC curves present the performance of a model over the full range of relative misclassification costs. AUROC and AUPR assess how models perform in aggregate over this range of relative misclassification costs, and thus, over a range of robotics contexts. Consequently, however, these metrics do not fully represent whether a model is appropriate in a *specific* context. To represent a specific context of interest, F_β scores can aggregate PR curves with a preference toward precision or

recall. In the effort to prioritize precision, as argued previously, $F_{0.5}$ scores are presented in this work.

C. Discussion: Misclassification Detection Accuracy

Misclassification detection accuracy (named A_{MD} to avoid confusion with segmentation accuracy) provides an intuitive understanding of misclassification detection performance by reporting the proportion of *all* pixels (the denominator considering the entire image grid) in either of the following "safe" states: $(\text{accurate}, \text{certain})$ or $(\text{inaccurate}, \text{uncertain})$; no preference between FP and FN is expressed.

Segmentation networks are typically one component of a larger system, and there are some robotics contexts where FP and FN are not drastically different in effect. For example, semantic localization can reject a FP using additional processing steps in the localization pipeline, i.e., RANSAC in geometric refinement. Also, in semantic mapping, multiple views of a location are available, which allow for additional processing, e.g., majority voting, to reduce the effect of errors. In both cases, where subsequent steps provide additional filtering, a FP is less detrimental to the broader system.

D. Discussion: Class Distribution

A characteristic of ROC and PR curves is their insensitivity to the class distribution, defined as the proportion of pixels that are accurate versus inaccurate. Each model will have a different semantic segmentation accuracy, and so the class distributions will vary. This means that ROC and PR curves provide helpful analysis on the ability to detect misclassification, independent of segmentation accuracy.

In the context of this work, however, if two models have the same misclassification detection performance, the model with the higher $\mathrm{p}(\text{accurate})$ should be chosen. More specifically, the proportion of pixels assigned, with certainty, to the known semantic classes should be considered, i.e., $\mathrm{p}(\mathrm{a}, \mathrm{c})$. In the interest of jointly considering misclassification detection and semantic segmentation performance, this work presents a procedure to describe both objectives intuitively.

Let us consider $F_{0.5}$ and A_{MD} versus $\mathrm{p}(\mathrm{a}, \mathrm{c})$. This allows us to determine the model and uncertainty threshold at which misclassification detection is performed best *and* the proportion of confidently segmented pixels returned at that threshold. These plots intuitively describe both the "introspectiveness" and the "usefulness" of the model. Note that the point on the curve at maximum $\mathrm{p}(\mathrm{a}, \mathrm{c})$ corresponds to the segmentation accuracy, as all pixels are treated as certain at this threshold, and so $\max[\mathrm{p}(\mathrm{a}, \mathrm{c})] = \mathrm{p}(\text{accurate})$.

VIII. Experimental Results

This section presents test metrics and qualitative examples discussing the benefits of our approach in Section VIII-K.

A. Source: Cityscapes, Target: SAX London

For each plot in the first row of Fig. 5, γ-SSL performs best. Its precision is higher for nearly all values of recall, with a corresponding increase of 19% and 8% from the best benchmark in AUROC and AUPR respectively (see Tables II and III). It has the highest values of $\mathrm{MaxA}_{\mathrm{MD}}$ and $\mathrm{MaxF}_{0.5}$, returning them at $\mathrm{p}(\mathrm{a}, \mathrm{c})$ exceeded only by $\mathrm{MCD}_{0.2}$ (however, at a lower value for $\mathrm{MaxA}_{\mathrm{MD}}$ and $\mathrm{MaxF}_{0.5}$).

The best-performing benchmarks are the $\mathrm{MCD}_{0.2}$ models, with MI outperforming PE. This is because they are both more introspective as judged by AUROC and AUPR, and also have a high segmentation accuracy. The latter suggests the dropout layers allowed for greater generalization to the SAX domains, compared with a standard segmentation network.

On average, the performance for this domain was relatively similar for the epistemic and representation-based methods in terms of AUROC and AUPR. However, while the representation-based methods exhibited similar $\mathrm{MaxA}_{\mathrm{MD}}$ and $\mathrm{MaxF}_{0.5}$ values to epistemic methods, they do so at lower $\mathrm{p}(\mathrm{a}, \mathrm{c})$ on average; thus, overall, they perform less well.

Note that γ-$\mathrm{SSL}_{\mathrm{iL}}$ for the SAX London, KITTI and BDD domains does not exist in the results. For SAX London γ-SSL and γ-$\mathrm{SSL}_{\mathrm{iL}}$ are the same model, and for the latter two, it is not clear that SAX London is an intermediate domain for these target domains; see Section VI-C for more details.

TABLE II
Misclassification Detection AUROC With Source: Cityscapes

		AUROC				
	Method	LDN	NF	SCOT	KITTI	BDD
Epistemic	Ens-PE_5	0.630	0.645	0.629	0.845	0.758
	Ens-MI_5	0.551	0.612	0.532	0.705	0.696
	Ens-PE_{10}	0.603	0.629	0.608	0.828	0.755
	Ens-MI_{10}	0.554	0.629	0.527	0.690	0.713
	MCD-$\mathrm{PE}_{0.2}$	0.727	0.739	0.735	0.864	0.801
	MCD-$\mathrm{MI}_{0.2}$	0.755	0.774	0.725	0.815	0.801
	MCD-DSL	0.697	0.742	0.676	0.855	0.761
	– Mean –	0.645	0.681	0.633	0.800	0.755
Representation	Softmax	0.723	0.686	0.674	0.785	0.706
	$\mathrm{Softmax}_{\mathrm{A}}$	0.722	0.685	0.672	0.784	0.705
	FeatDist	0.62	0.605	0.607	0.575	0.641
	$\mathrm{FeatDist}_{\mathrm{A}}$	0.645	0.642	0.585	0.585	0.648
	ViM	0.635	0.700	0.640	0.714	0.745
	DUM	0.483	0.565	0.510	0.501	0.502
	– Mean –	0.638	0.647	0.615	0.657	0.658
Ours	γ-SSL	**0.895**	0.880	0.776	**0.888**	**0.835**
	γ-$\mathrm{SSL}_{\mathrm{iL}}$	-	**0.880**	**0.859**	-	-

TABLE III
Misclassification Detection AUPR With Source: Cityscapes

		AUPR				
	Method	LDN	NF	SCOT	KITTI	BDD
Epistemic	Ens-PE_5	0.714	0.733	0.663	0.956	0.854
	Ens-MI_5	0.663	0.737	0.570	0.910	0.830
	Ens-PE_{10}	0.710	0.724	0.634	0.952	0.857
	Ens-MI_{10}	0.667	0.752	0.554	0.905	0.840
	MCD-$\mathrm{PE}_{0.2}$	0.851	0.859	0.737	0.926	0.816
	MCD-$\mathrm{MI}_{0.2}$	0.878	0.889	0.744	0.956	0.904
	MCD-DSL	0.785	0.818	0.602	**0.957**	0.851
	– Mean –	0.753	0.787	0.643	0.937	0.850
Representation	Softmax	0.711	0.631	0.525	0.840	0.686
	$\mathrm{Softmax}_{\mathrm{A}}$	0.711	0.631	0.524	0.840	0.686
	FeatDist	0.693	0.650	0.507	0.660	0.718
	$\mathrm{FeatDist}_{\mathrm{A}}$	0.721	0.699	0.456	0.637	0.727
	ViM	0.705	0.708	0.502	0.800	0.797
	DUM	0.630	0.622	0.350	0.367	0.319
	– Mean –	0.695	0.657	0.477	0.691	0.656
Ours	γ-SSL	**0.949**	0.921	0.726	0.951	**0.911**
	γ-$\mathrm{SSL}_{\mathrm{iL}}$	-	**0.942**	**0.887**	-	-

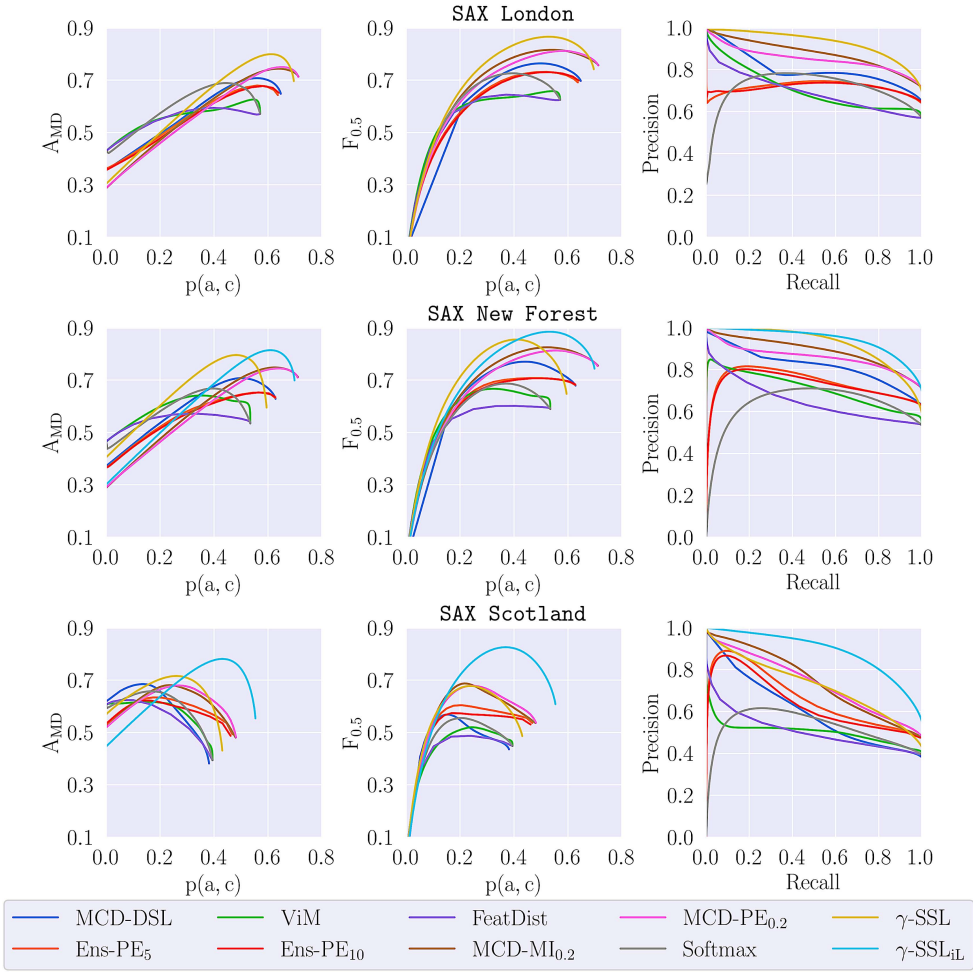


Fig. 5. For each SAX domain, a row of plots describes the misclassification detection performance of a series of benchmarks and the proposed methods, γ-SSL and γ-SSL$_{iL}$. Misclassification detection accuracy, A_{MD}, and F-score, $F_{0.5}$, aggregate performance into a single metric, where a larger value of each represents a more "introspective" model. They are plotted versus $p(a, c)$, the proportion of pixels that are accurate and certain, as this represents the amount of accurate and useful semantic information the model can extract from images; also a metric maximized by the ideal model. Note that the maximum value of $p(a, c)$ is equal to the segmentation accuracy, $\max[p(a, c)] = p(\text{accurate})$. Best viewed in color.

B. Source: Cityscapes, Target: SAX New Forest

In SAX New Forest, γ-SSL$_{iL}$ and γ-SSL perform similarly for AUROC and AUPR with γ-SSL$_{iL}$ having slightly higher AUPR (0.942 versus 0.921). The increases from γ-SSL to γ-SSL$_{iL}$ for MaxA_{MD} and $\text{MaxF}_{0.5}$ are modest at 2.4% and 3.5%, respectively, however the increases in $p(a, c)$ at which they occur are far larger at 25.9% and 30.7%. This coincides with a large increase in segmentation accuracy from γ-SSL to γ-SSL$_{iL}$, as seen from the maximum $p(a, c)$ in Fig. 5. For this domain, this confirms the hypothesis that presenting unlabeled target images in a curriculum improves semi-supervised learning both in segmentation quality and uncertainty estimation.

Compared to $\text{MCD}_{0.2}$, γ-SSL has higher AUROC and AUPR; however, the better method is context-dependent. The increase in MaxA_{MD} and $\text{MaxF}_{0.5}$ scores from $\text{MCD}_{0.2}$ to γ-SSL are 6.4% and 3.64%, respectively, but occur at a $p(a, c)$ 22.7% and 22.2% lower. This is a possible example of a model with higher segmentation accuracy (i.e., higher $p(a, c)$) preferable to a more introspective model (i.e., higher AUROC, AUPR); this demonstrates the benefit of presenting results in this way. This is not true for γ-SSL$_{iL}$, with an increase in MaxA_{MD} and $\text{MaxF}_{0.5}$ scores of 9.0% and 7.3% at a change in $p(a, c)$ of -2.7% and 1.7% compared to MCD-MI$_{0.2}$.

As in SAX London, the $\text{MCD}_{0.2}$ models were the best-performing benchmarks on each metric. In general, the epistemic methods outperformed the representation-based methods, with higher mean AUROC and AUPR. While the mean values of MaxA_{MD} and $\text{MaxF}_{0.5}$ were not significantly different, the values of $p(a, c)$ at which they occur were higher for epistemic methods than for representation methods.

C. Source: Cityscapes, Target: SAX Scotland

In this domain, the increase in performance from γ-SSL to γ-SSL$_{iL}$ is at its greatest, and the γ-SSL$_{iL}$ model far exceeds the performance of the other models, as in Fig. 5. The increase over the next best model for metrics AUROC, AUPR, $\text{MaxA}_{MD}@p(a, c)$, and $\text{MaxF}_{0.5}@p(a, c)$ are as follows: 10.7%, 19.2%, 9.1% @ 65.8%, and 20.2% @ 72.1%.

Once again, the performance increase in segmentation quality *and* uncertainty estimation can be attributed to the curriculum training procedure described in Section VI-C.

γ-SSL performs comparably to $\mathrm{MCD}_{0.2}$ in this domain, which as the distributional shift between SAX Scotland and Cityscapes so significant that it is challenging to use the semi-supervised task to improve the model's representation.

On average, epistemic methods outperform representation-based ones to a larger extent in this domain, characterized by a mean increase in AUPR of 34.8% from the latter to the former. This is because representation methods rely on a representation learned from Cityscapes, which is significantly distributionally shifted from SAX Scotland.

D. Effect of Increasing Distributional Shift

As discussed in Section VI-A, the following domains are in order of ascending distributional shift: London, New Forest, and Scotland, as evidenced by the corresponding reduction in segmentation accuracy—0.571, 0.538, 0.394—for a network trained solely on Cityscapes. Table X shows that the method type has an effect on the extent to which misclassification detection performance degrades as distributional shift increases.

1) Epistemic Methods: For all epistemic methods, AUROC and AUPR increase from London to New Forest—6.0% and 5.0% on average, respectively. This suggests that these techniques perform better uncertainty estimation as the proportion of errors related to distributional shift increases, which aligns with the stated motivation of the methods. However, as distributional uncertainty significantly increases, not every method improves further. In fact, both AUROC and AUPR significantly decrease from New Forest to Scotland, -7.4% and -18.3% on average. This suggests that there is a limit beyond which these methods start to degrade significantly, as was also reported for a range of epistemic uncertainty estimation methods in [53].

2) Representation Methods: In general, these methods are less robust to distributional shift than the epistemic ones, with a difference in AUPR of -5.4% and -27.3% for London to New Forest, and New Forest to Scotland, respectively. The changes were less uniform for AUROC; however, the majority of models tested decreased in performance for *both* shifts. In addition, these methods have a significant reduction in the $\mathrm{p(a, c)}$ at which $\mathrm{MaxA_{MD}}$ and $\mathrm{MaxF_{0.5}}$ occur compared with epistemic methods.

3) γ-SSL Methods: Much like many of the representation methods, the performance of γ-SSL decreases from London to New Forest, and again to Scotland. γ-SSL decreases less on average in terms of AUPR, but more in terms of AUROC. Independent of the increase in segmentation accuracy, the use of a curriculum for γ-$\mathrm{SSL_{iL}}$ reduces the effects of large distributional shifts on uncertainty estimation, as evidenced by the γ-$\mathrm{SSL_{iL}}$ having by far the smallest reduction in AUPR from New Forest to Scotland. Factoring in the increased segmentation accuracy, γ-$\mathrm{SSL_{iL}}$ also exhibits a much smaller reduction in the value of $\mathrm{p(a, c)}$ at which $\mathrm{MaxA_{MD}}$ and $\mathrm{MaxF_{0.5}}$ occur when compared to γ-SSL.

TABLE IV
MAXIMUM ACCURACY $\mathrm{A_{MD}}$ AND $\mathrm{p(a, c)}$ WITH SOURCE: CITYSCAPES

		$\mathrm{MaxA_{MD}}$ @ $\mathrm{p(a, c)}$				
	Method	LDN	NF	SCOT	KITTI	BDD
Epistemic	Ens-$\mathrm{PE_5}$	0.679 @ 0.560	0.652 @ 0.581	0.634 @ 0.175	0.818 @ 0.751	0.734 @ 0.607
	Ens-$\mathrm{MI_5}$	0.645 @ 0.620	0.643 @ 0.613	0.580 @ 0.076	0.798 @ 0.797	0.692 @ 0.668
	Ens-$\mathrm{PE_{10}}$	0.678 @ 0.582	0.653 @ 0.564	0.622 @ 0.142	0.812 @ 0.768	0.728 @ 0.602
	Ens-$\mathrm{MI_{10}}$	0.654 @ 0.622	0.650 @ 0.611	0.578 @ 0.079	0.798 @ 0.798	0.698 @ 0.624
	MCD-$\mathrm{PE_{0.2}}$	0.750 @ 0.646	0.745 @ 0.645	0.679 @ 0.270	0.849 @ 0.792	0.765 @ 0.608
	MCD-$\mathrm{MI_{0.2}}$	0.744 @ 0.640	0.748 @ 0.625	0.681 @ 0.234	0.833 @ 0.823	0.754 @ 0.607
	MCD-DSL	0.708 @ 0.559	0.708 @ 0.503	0.685 @ 0.136	0.830 @ 0.745	0.738 @ 0.571
	– Mean –	0.694 @ 0.604	0.686 @ 0.592	0.637 @ 0.159	0.795 @ 0.782	0.730 @ 0.612
Representation	Softmax	0.689 @ 0.444	0.668 @ 0.396	0.657 @ 0.164	0.718 @ 0.434	0.692 @ 0.479
	$\mathrm{Softmax_A}$	0.689 @ 0.446	0.668 @ 0.401	0.656 @ 0.167	0.717 @ 0.435	0.692 @ 0.480
	FeatDist	0.594 @ 0.371	0.572 @ 0.334	0.624 @ 0.055	0.588 @ 0.574	0.620 @ 0.489
	$\mathrm{FeatDist_A}$	0.617 @ 0.449	0.603 @ 0.378	0.636 @ 0.032	0.564 @ 0.384	0.626 @ 0.502
	ViM	0.626 @ 0.546	0.641 @ 0.353	0.617 @ 0.153	0.655 @ 0.386	0.687 @ 0.480
	DUM	0.625 @ 0.625	0.618 @ 0.556	0.649 @ 0.000	0.733 @ 0.732	0.637 @ 0.635
	– Mean –	0.640 @ 0.480	0.628 @ 0.403	0.640 @ 0.095	0.663 @ 0.491	0.659 @ 0.511
Ours	γ-SSL	**0.83** @ 0.625	0.796 @ 0.483	0.716 @ 0.260	**0.856** @ 0.767	**0.770** @ 0.568
	γ-$\mathrm{SSL_{iL}}$	-	**0.815** @ 0.608	**0.781** @ 0.431	-	-

This represents the most conservative point of the system, given that the number of 'safe' pixels is maximised.

TABLE V
MAXIMUM $\mathrm{F_{0.5}}$ SCORE AND $\mathrm{p(a, c)}$ WITH SOURCE: CITYSCAPES

		$\mathrm{MaxF_{0.5}}$ @ $\mathrm{p(a, c)}$				
	Method	LDN	NF	SCOT	KITTI	BDD
Epistemic	Ens-$\mathrm{PE_5}$	0.732 @ 0.520	0.708 @ 0.460	0.604 @ 0.206	0.889 @ 0.613	0.801 @ 0.496
	Ens-$\mathrm{MI_5}$	0.694 @ 0.609	0.691 @ 0.605	0.528 @ 0.461	0.832 @ 0.795	0.758 @ 0.472
	Ens-$\mathrm{PE_{10}}$	0.730 @ 0.539	0.707 @ 0.517	0.574 @ 0.168	0.880 @ 0.608	0.795 @ 0.492
	Ens-$\mathrm{MI_{10}}$	0.702 @ 0.603	0.696 @ 0.597	0.522 @ 0.460	0.833 @ 0.761	0.767 @ 0.488
	MCD-$\mathrm{PE_{0.2}}$	0.812 @ 0.584	0.813 @ 0.559	0.678 @ 0.248	0.914 @ 0.686	0.836 @ 0.530
	MCD-$\mathrm{MI_{0.2}}$	0.816 @ 0.541	0.825 @ 0.523	0.687 @ 0.215	0.891 @ 0.669	0.836 @ 0.500
	MCD-DSL	0.764 @ 0.499	0.770 @ 0.441	0.568 @ 0.155	0.900 @ 0.644	0.802 @ 0.491
	– Mean –	0.750 @ 0.556	0.744 @ 0.529	0.594 @ 0.273	0.877 @ 0.682	0.799 @ 0.496
Representation	Softmax	0.726 @ 0.395	0.687 @ 0.370	0.555 @ 0.205	0.777 @ 0.366	0.735 @ 0.429
	$\mathrm{Softmax_A}$	0.726 @ 0.397	0.687 @ 0.373	0.554 @ 0.207	0.777 @ 0.366	0.735 @ 0.430
	FeatDist	0.645 @ 0.371	0.601 @ 0.408	0.487 @ 0.240	0.640 @ 0.571	0.672 @ 0.434
	$\mathrm{FeatDist_A}$	0.672 @ 0.360	0.643 @ 0.296	0.449 @ 0.240	0.612 @ 0.525	0.680 @ 0.428
	ViM	0.658 @ 0.543	0.667 @ 0.326	0.520 @ 0.243	0.721 @ 0.324	0.735 @ 0.391
	DUM	0.676 @ **0.625**	0.664 @ 0.528	0.419 @ 0.315	0.774 @ 0.731	0.686 @ 0.635
	– Mean –	0.684 @ 0.449	0.658 @ 0.384	0.497 @ 0.242	0.717 @ 0.481	0.707 @ 0.458
Ours	γ-SSL	**0.893** @ 0.548	0.855 @ 0.407	0.678 @ 0.239	**0.920** @ 0.676	**0.843** @ 0.475
	γ-$\mathrm{SSL_{iL}}$	-	**0.885** @ 0.532	**0.826** @ 0.370	-	-

E. Source: Cityscapes, Target: KITTI and BDD

The results for KITTI and BDD can be found in Tables II–V. The accuracy of γ-SSL for KITTI, SAX London, BDD, and SAX New Forest are as follows: 0.817, 0.703, 0.684, and 0.595; therefore, KITTI is the least distributionally shifted domain w.r.t. Cityscapes, and BDD is more distributionally shifted than SAX London.

1) KITTI: The AUROC, $\mathrm{MaxA_{MD}}$, and $\mathrm{MaxF_{0.5}}$ metrics for our γ-SSL method exceeded that of all of the benchmarks, with only AUPR exceeded by the MCD-DSL method. Epistemic methods significantly outperformed representation-based methods, as epistemic method outperformed the latter on almost all of the metrics. This experiment, therefore, provides extra evidence that our method performs well in target domains close to the source domain.

2) BDD: γ-SSL is the best performing model for each metric with BDD as the target domain, with epistemic methods on average outperforming the representation-based methods. Despite being less distributionally shifted than SAX New Forest according to segmentation accuracy, each of the uncertainty estimation metrics are lower for BDD than SAX New Forest in a break from the trend. The hypothesis for why this is that there is significantly more diversity in BDD compared with SAX New Forest, and learning a representation for uncertainty estimation is more difficult in a more diverse domain.

F. Source: BDD

In order to investigate the generality of this approach, this work also conducts secondary experiments with BDD as the

TABLE VI
MISCLASSIFICATION DETECTION AUROC WITH SOURCE: BDD

		AUROC		
	Method	LDN	NF	SCOT
Epistemic	Ens-PE_5	0.781	0.854	0.862
	Ens-MI_5	0.744	0.809	0.818
	Ens-PE_{10}	0.785	0.854	**0.866**
	Ens-MI_{10}	0.761	0.827	0.861
	$\text{MCD-PE}_{0.2}$	0.834	0.841	0.822
	$\text{MCD-MI}_{0.2}$	0.808	0.816	0.739
	MCD-DSL	0.821	0.803	0.805
	– Mean –	0.791	0.829	0.825
Representation	Softmax	0.825	0.819	0.810
	Softmax_A	0.825	0.819	0.810
	FeatDist	0.605	0.599	0.563
	FeatDist_A	0.625	0.631	0.598
	ViM	0.663	0.664	0.560
	DUM	0.480	0.431	0.398
	– Mean –	0.671	0.661	0.623
Ours	γ-SSL	**0.889**	0.876	0.833
	$\gamma\text{-SSL}_\text{iL}$	-	**0.882**	0.855

TABLE VII
MISCLASSIFICATION DETECTION AUPR WITH SOURCE: BDD

		AUPR		
	Method	LDN	NF	SCOT
Epistemic	Ens-PE_5	0.896	0.941	0.885
	Ens-MI_5	0.876	0.915	0.837
	Ens-PE_{10}	0.899	**0.942**	0.880
	Ens-MI_{10}	0.884	0.925	0.870
	$\text{MCD-PE}_{0.2}$	0.903	0.881	0.798
	$\text{MCD-MI}_{0.2}$	0.880	0.863	0.728
	MCD-DSL	0.902	0.914	0.849
	– Mean –	0.891	0.912	0.835
Representation	Softmax	0.901	0.904	0.816
	Softmax_A	0.901	0.905	0.816
	FeatDist	0.724	0.731	0.547
	FeatDist_A	0.758	0.770	0.604
	ViM	0.799	0.797	0.547
	DUM	0.549	0.530	0.321
	– Mean –	0.772	0.773	0.608
Ours	γ-SSL	**0.936**	0.935	0.858
	$\gamma\text{-SSL}_\text{iL}$	-	0.928	**0.900**

TABLE VIII
MAXIMUM ACCURACY A_MD AND $\text{p}(\text{a}, \text{c})$ WITH SOURCE: BDD

		MaxA_MD @ $\text{p}(\text{a}, \text{c})$		
	Method	LDN	NF	SCOT
Epistemic	Ens-PE_5	0.739 @ 0.630	0.783 @ 0.621	0.790 @ 0.347
	Ens-MI_5	0.716 @ 0.623	0.766 @ 0.637	0.753 @ 0.343
	Ens-PE_{10}	0.741 @ 0.619	0.783 @ 0.627	0.792 @ 0.331
	Ens-MI_{10}	0.727 @ 0.619	0.774 @ 0.629	**0.797** @ 0.351
	$\text{MCD-PE}_{0.2}$	0.753 @ 0.488	0.760 @ 0.390	0.766 @ 0.217
	$\text{MCD-MI}_{0.2}$	0.729 @ 0.454	0.742 @ 0.362	0.729 @ 0.176
	MCD-DSL	0.756 @ 0.549	0.761 @ 0.638	0.727 @ 0.387
	– Mean –	0.737 @ 0.569	0.767 @ 0.558	0.765 @ 0.307
Representation	Softmax	0.761 @ 0.553	0.760 @ 0.573	0.755 @ 0.275
	Softmax_A	0.761 @ 0.554	0.761 @ 0.574	0.754 @ 0.275
	FeatDist	0.673 @ 0.609	0.677 @ 0.674	0.603 @ 0.154
	FeatDist_A	0.706 @ 0.646	0.727 @ 0.668	0.590 @ 0.265
	ViM	0.662 @ 0.662	0.683 @ 0.608	0.588 @ 0.114
	DUM	0.562 @ 0.562	0.596 @ 0.596	0.604 @ 0.000
	– Mean –	0.688 @ 0.598	0.701 @ 0.616	0.649 @ 0.181
Ours	γ-SSL	**0.818** @ 0.599	0.805 @ 0.567	0.762 @ 0.331
	$\gamma\text{-SSL}_\text{iL}$	-	**0.809** @ 0.578	0.769 @ 0.417

This represents the most conservative point of the system, given that the number of 'safe' pixels is maximised.

TABLE IX
MAXIMUM $\text{F}_{0.5}$ SCORE AND $\text{p}(\text{a}, \text{c})$ WITH SOURCE: BDD

		$\text{MaxF}_{0.5}$ @ $\text{p}(\text{a}, \text{c})$		
	Method	LDN	NF	SCOT
Epistemic	Ens-PE_5	0.810 @ 0.488	0.872 @ 0.500	0.829 @ 0.307
	Ens-MI_5	0.792 @ 0.484	0.839 @ 0.528	0.777 @ 0.290
	Ens-PE_{10}	0.814 @ 0.488	0.872 @ 0.500	0.825 @ 0.284
	Ens-MI_{10}	0.802 @ 0.483	0.855 @ 0.519	0.815 @ 0.309
	$\text{MCD-PE}_{0.2}$	0.833 @ 0.403	0.814 @ 0.328	0.748 @ 0.199
	$\text{MCD-MI}_{0.2}$	0.820 @ 0.380	0.803 @ 0.311	0.685 @ 0.163
	MCD-DSL	0.829 @ 0.460	0.839 @ 0.525	0.769 @ 0.319
	– Mean –	0.814 @ 0.455	0.842 @ 0.459	0.778 @ 0.267
Representation	Softmax	0.833 @ 0.465	0.835 @ 0.483	0.764 @ 0.241
	Softmax_A	0.833 @ 0.468	0.836 @ 0.484	0.764 @ 0.241
	FeatDist	0.730 @ 0.564	0.731 @ 0.566	0.534 @ 0.191
	FeatDist_A	0.755 @ 0.614	0.778 @ 0.621	0.592 @ 0.285
	ViM	0.732 @ 0.433	0.747 @ 0.541	0.511 @ 0.214
	DUM	0.616 @ 0.562	0.648 @ 0.596	0.451 @ 0.396
	– Mean –	0.750 @ 0.518	0.763 @ 0.549	0.603 @ 0.261
Ours	γ-SSL	**0.887** @ 0.515	0.873 @ 0.495	0.797 @ 0.284
	$\gamma\text{-SSL}_\text{iL}$	-	**0.885** @ 0.501	**0.831** @ 0.348

source domain. The suite of results for this are found: Tables VI–IX. For both epistemic- and representation-based methods, using BDD as the source domain improves uncertainty estimation performance across all metrics. This benefit typically increases as the domain shift increases, e.g., for $\text{MaxF}_{0.5}$ the average percentage increases from Cityscapes to BDD for the benchmarks are: 9.1%, 14.6%, and 26.2% for SAX London, New Forest, and Scotland, respectively. The BDD labeled dataset is larger and more diverse, therefore, leading to a more general representation of the semantic classes.

1) SAX London: For this target domain, γ-SSL performs the best on each metric, with a 6.5% increase in $\text{MaxF}_{0.5}$ over the best performing benchmark. Across the board, the results for γ-SSL with Cityscapes as the source domain are better than in this experiment. The higher uncertainty estimation metrics coincide with a higher accuracies using Cityscapes as the source domain than BDD (0.703 and 0.688, respectively), suggesting that Cityscapes could be more similar to this domain than BDD.

2) SAX New Forest: The accuracy of the γ-SSL models are 0.666 and 0.595 for BDD and Cityscapes, respectively, suggesting a larger domain shift between Cityscapes and SAX New Forest, than for BDD and SAX New Forest. In all but AUPR, both the γ-SSL and $\gamma\text{-SSL}_\text{iL}$ models outperform all benchmarks. The results for using BDD as source are similar or slightly better than using Cityscapes. Better uncertainty estimation results for BDD would be predicted by the lesser domain shift, however this is not consistently shown. This is perhaps because BDD is more diverse, and a narrower definition of the source domain allows a greater separation of source and target, and simpler uncertainty estimation.

3) SAX Scotland: The results for γ-SSL are significantly better with source domain as BDD, than as Cityscapes, accompanied by similar but smaller improvements for $\gamma\text{-SSL}_\text{iL}$. Segmentation accuracies for γ-SSL of 0.495 and 0.431 for BDD and Cityscapes, respectively, suggest that the magnitude of the domain shifts would explain this. It is, however, also true in this experiment that the epistemic methods are significantly better resulting in comparable performance between $\gamma\text{-SSL}_\text{iL}$

Fig. 6. Box plot representing the achieved A_{MD} by calculating the uncertainty threshold with varying numbers of validation examples for a γ-SSL model. The dashed lines represent the values of A_{MD} achieved when using the entire test dataset to calculate the optimal threshold, before then testing on it.

and these methods for these metrics, i.e., which method is better depends on the metric considered.

The results demonstrate that while a different source domain has an effect on the quality of uncertainty estimation, our method still exceeds or is competitive with the results of the best benchmarks considered, while still having the low latency required for robotic deployment.

G. Optimal Threshold Calculation Testing

It is clear from Fig. 5 that there typically exists a threshold such that accurate and inaccurate pixels are optimally separated (according to A_{MD} and $F_{0.5}$). Calculation of this threshold typically requires a set of validation images from the test domain, which reduces the number of images available for testing. In addition, if this small set of validation images is not representative of the test dataset, then the calculated threshold will result in significantly worse misclassification detection performance.

The effect of the number of validation examples on misclassification detection performance for the γ-SSL models is investigated by calculating the metrics discussed in Section VII-A for a given withheld validation set on the remaining test images. The results for this are averaged over 100 trials, where for each the withheld validation set is selected randomly. Given that the method in this work calculates a threshold parameter during training, it is also possible for it to use this threshold during testing, such that *zero* validation examples are required. The mean and variability of these metrics is presented as a box plot in Fig. 6.

First, these plots demonstrate that the smaller the number of chosen validation images, the less well they represent the test dataset; therefore, the more variable the test performance. More importantly, they show that there is a minimal decrease in performance between using 20 validation images and calculating the threshold with 0, as per Section V-B. This means the γ-SSL methods are successfully able to calculate an appropriate threshold for the misclassification detection task without using *any* validation examples.

H. Cross-Domain Threshold Testing

The aim of this work is to propose a model that can estimate its mistakes with a feature-space distance threshold as the data distribution changes. It is, therefore, important that this threshold calculated for one domain is effective for all domains, rather than needing a specifically optimal threshold for each domain.

In order to investigate this (see Table XI), we compare the $F_{0.5}$ results for the optimal threshold value for a test domain and the value calculated from the target domain the model is trained on, e.g., for γ-SSL_{iL}-LDN (trained on unlabeled SAX London data), the threshold corresponding to the maximum $F_{0.5}$ score is used testing on the SAX New Forest and Scotland datasets and the corresponding $F_{0.5}$ scores are shown.

Table XI shows that a threshold optimal for one domain degrades performance only very slightly for another domain.

I. WildDash Results

So far in this work, the γ-SSL and γ-SSL_{iL} models have been trained on the SAX domains with operating conditions different to that of Cityscapes, and tested on these same SAX domains. In this section, the models are also tested on the WildDash dataset [54], in order to investigate how the models generalize

TABLE X
AUROC AND AUPR PERCENTAGE CHANGE AT INCREASING DISTRIBUTIONAL SHIFT, %ΔROC, AND %ΔPR, RESPECTIVELY

		LDN → NF		NF → SCOT	
	Method	%ΔROC	%ΔPR	%ΔROC	%ΔPR
Epistemic	Ens-PE_5	2.3	2.7	-2.5	-9.6
	Ens-MI_5	11.1	11.2	-13.0	-22.7
	Ens-PE_{10}	4.2	2.0	-3.2	-12.4
	Ens-MI_{10}	13.7	12.8	-16.3	-26.3
	MCD-$PE_{0.2}$	1.8	0.9	-0.6	-14.2
	MCD-$MI_{0.2}$	2.6	1.3	-6.4	-16.4
	MCD-DSL	6.5	4.2	-8.9	-26.4
	– Mean –	6.0	5.0	-7.3	-18.3
Representation	Softmax	-5.1	-11.2	-1.8	-16.8
	$Softmax_A$	-5.1	-11.2	-1.9	-17.0
	FeatDist	-2.3	-6.1	0.3	-22.1
	$FeatDist_A$	-0.4	-3.1	-9.0	-34.8
	ViM	10.3	0.5	-8.5	-29.2
	DUM	16.9	-1.4	-9.7	-43.6
	– Mean –	2.4	-5.4	-5.1	-27.3
Ours	γ-SSL	-1.7	-3.0	-11.8	-21.2
	γ-SSL_{iL}	-1.7	-0.7	-2.4	-5.8

A negative value represents a decrease in misclassification detection performance as distributional shift increases.

TABLE XI
CROSS-DOMAIN THRESHOLD TESTING RESULTS

(a) γ-SSL_{iL}-LDN

	$MaxF_{0.5}$	$F_{0.5}$ with γ-LDN	Δ
LDN	0.8930	0.8930	0%
NF	0.8827	0.8822	−0.058%
SCOT	0.7853	0.7839	−0.175%

(b) γ-SSL_{iL}-NF

	$MaxF_{0.5}$	$F_{0.5}$ with γ-NF	Δ
LDN	0.8834	0.8832	−0.017%
NF	0.8849	0.8849	0%
SCOT	0.7952	0.7949	−0.038%

(c) γ-SSL_{iL}-SCOT

	$MaxF_{0.5}$	$F_{0.5}$ with γ-SCOT	Δ
LDN	0.8768	0.8764	−0.038%
NF	0.8806	0.8805	−0.007%
SCOT	0.8257	0.8257	0%

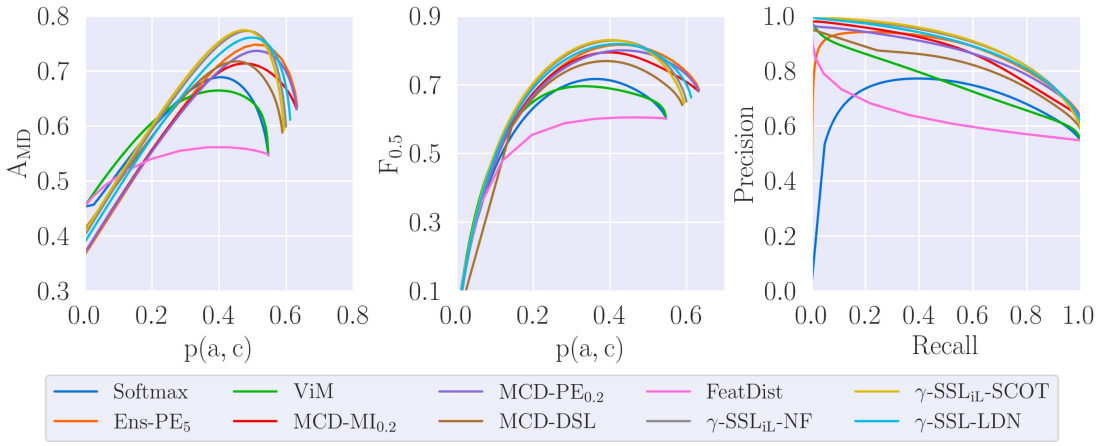


Fig. 7. Misclassification detection results on the WildDash dataset [54]. γ-SSL-LDN refers to a γ-SSL trained on the SAX London unlabeled dataset, whereas γ-SSL_{iL}-NF, γ-SSL_{iL}-SCOT refer to γ-SSL_{iL} models that are trained on the SAX New Forest and SAX Scotland unlabeled datasets, while also using SAX London as part of a curriculum. Best viewed in color.

TABLE XII
TIMING RESULTS

Method	GPU [Hz]	CPU [Hz]
Vanilla	159.12	1.53
MCD	19.62	0.46
Ens-5^{LM}	5.27	0.75
Ens-5^{HM}	27.22	0.75
Ens-10^{LM}	2.99	0.38
Ens-10^{HM}	16.64	0.38
Ours	183.37	1.52

to a test dataset with different operating conditions to both the labeled *and* unlabeled training data. This dataset does not define a single domain but includes images from a diverse set of domains, including: different weather conditions, day/night, and geographic locations from across the world. Misclassification detection performance on this dataset is, therefore, a measure of how well these models can detect error due to OoD instances unlike anything seen in the labeled or unlabeled training datasets, or how specific they are to the domain of the unlabeled training data.

As shown in Fig. 7, γ-SSL_{iL}-SCOT outperform all benchmarks, with an AUROC and AUPR of 0.852 and 0.896, compared with the best benchmark, Ens-PE_5, returning 0.803 and 0.868.

First, this demonstrates that although the γ-SSL_{iL} models have been trained to mitigate error in specific operating conditions based on geographic location, they can also effectively detect error due to never-before-seen conditions. Second, given that these values for γ-SSL_{iL}-SCOT are lower than the values for the SAX test domains, it also demonstrates that the best performance is reached when an unlabeled training dataset is collected from the same domain as the test data.

J. Timing Results

In Table XII, the frequency at which each method can operate is presented on differing hardware, as a low latency is a key characteristic for robotic perception systems. The GPU tested upon is a NVIDIA V100 GPU, while the CPU is on Macbook Pro with M2 Pro CPU. The Vanilla method is a DeepLabV3+ segmentation network [47] detailed in Section VI-B, with the difference from Ours being that it uses a convolutional layer instead of prototype segmentation, and therefore, also does not contain a projection network. The timing results for Vanilla, therefore, represent all of the representation-based methods described in in Section VI-D. The MCD and Ens methods are the same as those described in Section VI-D, (i.e., five and ten member ensembles and eight samples for the MCD method). The superscript LM and HM relate to different inference methods for the ensembles, with the former (low memory) only loading one network into GPU memory at a time, while the latter (high memory) loads every member network at once, while still performing inference sequentially.

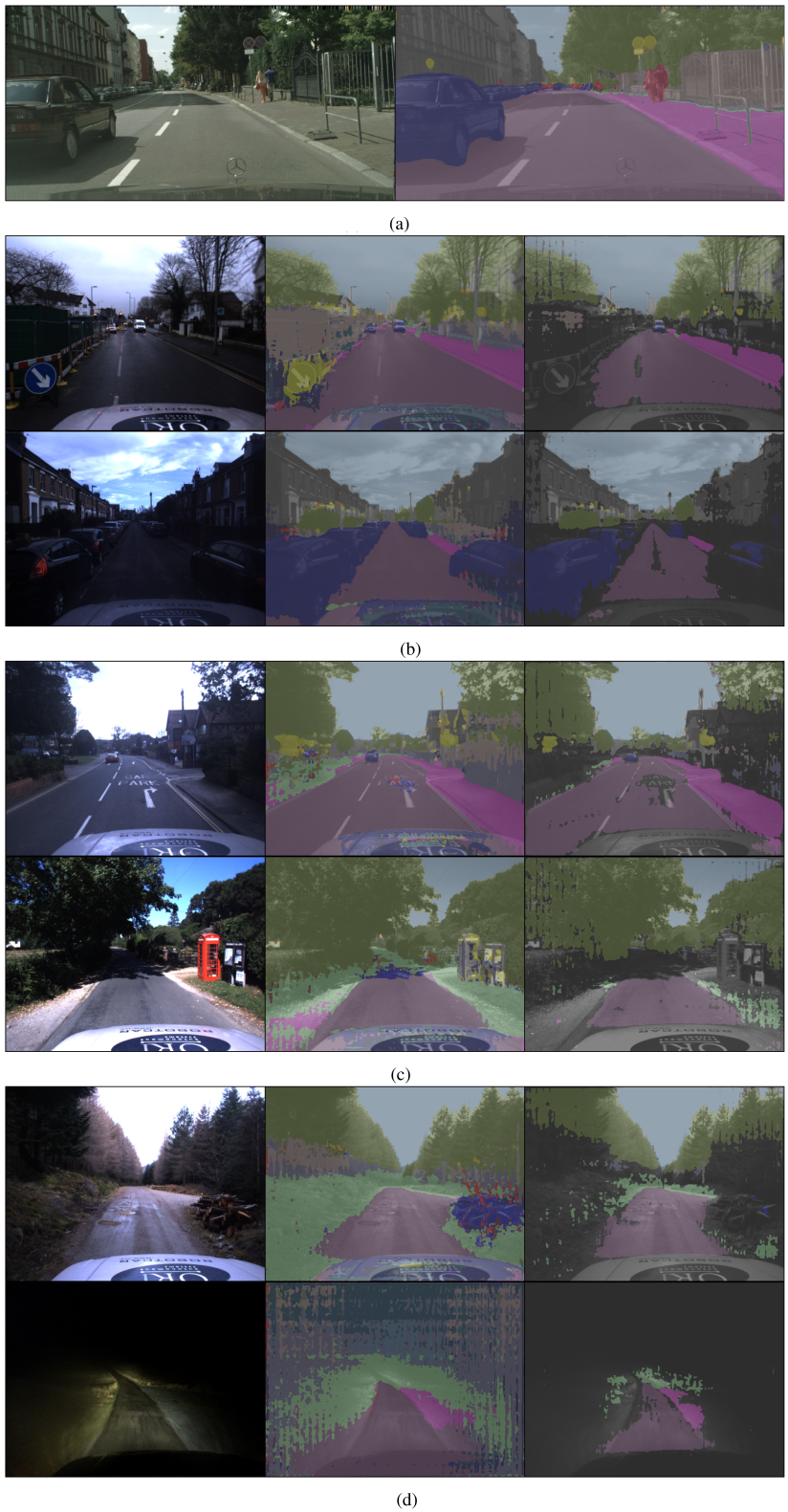
These results demonstrate that our proposed method operates at significantly lower latency than the MCD and ensemble methods, and is no slower than the segmentation network from which our method is built.

K. Qualitative Results

Presentation of qualitative results can be found in Fig. 8. Again, labels are only available in the source domain—Cityscapes—for which segmentations are very high quality. The segmentation performance degrades with increasing domain shift as expected from London → New Forest → Scotland. Correspondingly, however, uncertainty mitigates these erroneous predictions. Consider several examples from these samples as follows.

1) The lane-obstructed street sign in the first London example, not among the known signs in the Cityscapes dataset.
2) The telephone box in the second New Forest example (not in the Cityscapes domain or list of known classes).
3) The pile of timber in the first Scotland example—classified as vehicle.

All of these are correctly assigned high uncertainty.



(a)

(b)

(c)

(d)

Fig. 8. Qualitative results for Cityscapes and the SAX domains. As the SAX RGB images (left) become more dissimilar from Cityscapes (from top to bottom), the corresponding semantic segmentations (center) decrease in quality. *However*, for these poorly segmented regions, high uncertainty is largely expressed over them, shown in black (right). (a) Cityscapes. (b) London. (c) New Forest. (d) Scotland.

TABLE XIII
ABLATIONS MISCLASSIFICATION DETECTION

	Method	AUROC LDN	AUROC NF	AUROC SCOT	AUPR LDN	AUPR NF	AUPR SCOT
Ablations	NoSSL	0.803	0.774	0.72	0.872	0.829	0.686
	NoSAX	0.793	0.771	0.669	0.824	0.794	0.455
	$M_{\gamma=-\infty}$	0.805	0.761	0.711	0.9	0.827	0.646
	Sym-Param	0.851	0.598	0.622	0.8	0.539	0.486
	Sym-Non-Param	0.643	0.643	0.621	0.382	0.235	0.165
	NoRegL	0.815	0.827	0.72	0.882	0.847	0.621
	MCD-SSL	0.776	0.81	0.705	0.794	0.857	0.592
Ours	γ-SSL	**0.895**	0.88	0.776	**0.949**	0.921	0.726
	γ-SSL$_{\text{iL}}$	-	**0.88**	**0.859**	-	**0.942**	**0.887**

(a)

	Method	MaxA_{MD} @ p(a, c) LDN	NF	SCOT
Ablations	NoSSL	0.727 @ 0.482	0.706 @ 0.424	0.680 @ 0.204
	NoSAX	0.729 @ 0.465	0.717 @ 0.463	0.653 @ 0.003
	$M_{\gamma=-\infty}$	0.754 @ 0.586	0.692 @ 0.426	0.685 @ 0.174
	Sym-Param	0.790 @ 0.253	0.596 @ 0.368	0.756 @ 0.053
	Sym-Non-Param	0.752 @ 0.015	~~0.864 @ 0.000~~	~~0.902 @ 0.000~~
	NoRegL	0.766 @ 0.552	0.748 @ 0.391	0.665 @ 0.201
	MCD-SSL	0.747 @ 0.594	0.748 @ 0.482	0.707 @ 0.153
Ours	γ-SSL	**0.83** @ 0.625	0.796 @ 0.483	0.716 @ 0.260
	γ-SSL$_{\text{iL}}$	-	**0.815** @ 0.608	**0.781** @ 0.431

Struck through results are discounted as no pixels that are both accurate and certain are found.

(b)

	Method	$\text{MaxF}_{0.5}$ @ p(a, c) LDN	NF	SCOT
Ablations	NoSSL	0.790 @ 0.380	0.753 @ 0.364	0.626 @ 0.203
	NoSAX	0.769 @ 0.401	0.756 @ 0.403	0.505 @ 0.220
	$M_{\gamma=-\infty}$	0.826 @ 0.492	0.746 @ 0.353	0.602 @ 0.184
	Sym-Param	0.731 @ 0.219	0.608 @ 0.381	0.495 @ 0.057
	Sym-Non-Param	0.394 @ 0.088	0.281 @ 0.030	0.210 @ 0.017
	NoRegL	0.806 @ 0.507	0.781 @ 0.332	0.569 @ 0.241
	MCD-SSL	0.809 @ 0.518	0.804 @ 0.416	0.589 @ 0.159
Ours	γ-SSL	**0.893** @ 0.548	0.855 @ 0.407	0.678 @ 0.239
	γ-SSL$_{\text{iL}}$	-	**0.885** @ 0.532	**0.826** @ 0.370

(c)

IX. ABLATION STUDIES

This section investigates the effect of each component of the system on misclassification-detection performance by removing key aspects of the system during training. Results from these ablation experiments can be found in Table XIII(a) and Fig. 9.

A. Is Distance to Prototypes a Good Measure of Uncertainty?

This is investigated by updating the model using only the source images through the supervised loss L_{s}, i.e., no semi-supervised learning takes place. In this way, the mechanism for generating uncertainty estimates, i.e., calculating the cosine similarity with the prototypes, can be investigated. Notably, except for DUM, the same model weights are shared between our method and the representation-based benchmarks. Comparing the results in Tables II–XIII(a), the ablation, named NoSSL, outperforms the other representation-based methods in AUROC and AUPR .

These results support our argument that the method for calculating M_γ is a good inductive bias and helps to set up the positive feedback loop discussed in Section V. The fact that NoSSL performs significantly worse than γ-SSL and γ-SSL$_{\text{iL}}$ for each metric suggests that the method's success relies on the representation learned on unlabeled target domain data.

B. Is the Target Domain Data Required?

In this investigation, Cityscapes data are used for the semi-supervised task instead of the SAX unlabeled data (hence, is called NoSAX). The objective is to determine whether the proposed method is leveraging the unlabeled target domain data or whether our uncertainty estimation results are due only to the semi-supervised task and objectives.

Compared with γ-SSL and γ-SSL$_{\text{iL}}$, the AUROC and AUPR results in Table XIII show a worse misclassification detection performance for NoSAX on the SAX test datasets. This confirms the utility of collecting large, diverse datasets containing near-distribution and OoD instances, as this ablation empirically shows that using such a dataset during training improves the detection of OoD instances during testing.

C. Is M_γ Required to Learn Uncertainty Estimation?

This experiment, named $\text{M}_{\gamma=-\infty}$, trains a model by maximising the consistency between *all* pixels. This, therefore, investigates the loss function in (8) and whether standard semi-supervised training is sufficient to learn a good representation for uncertainty estimation.

The results consistently show that the proposed γ-SSL performs better at misclassification detection on each metric. This suggests that attenuating the loss for uncertain pixels facilitates learning a representation suitable for uncertainty estimation. In addition, the results in Fig. 9 show that the segmentation accuracy is lower for this ablation (remembering that $\max[\text{p}(\text{a}, \text{c})]$ is the segmentation accuracy), showing it is beneficial to use M_γ to filter out noise in the semi-supervised consistency task introduced by the challenging, uncurated nature of the unlabeled training images.

D. Does Branch Asymmetry Prevent Feature Collapse?

Sym-Non-Param and Sym-Param produce segmentations using only nonparametric prototype segmentation and a parameterized segmentation head, respectively, and are, thus, both symmetric, unlike our system (see Section V-E).

For both methods, the models nearly always suffered feature collapse, where each feature is embedded near a single class prototype (Road in this case). The exception is for Sym-Proto on the SAX New Forest dataset. When collapse occurs, the key observation is that segmentation accuracy greatly deteriorates, even if AUROC and AUPR look acceptable. First, this ablation provides evidence that the asymmetric branches successfully prevent this type of failure. Second, it confirms that looking at AUROC and AUPR alone is clearly insufficient to fully evaluate the model, and that integrating segmentation quality into the metrics is useful.

E. Do the Additional Losses Provide Useful Regularization?

By removing both L_{u} and L_{p} for NoRegL, the effect of these additional losses can be investigated. The result is not a complete feature collapse but a deterioration of misclassification-detection and segmentation performance on every metric. This suggests that by spreading out features and prototypes on the

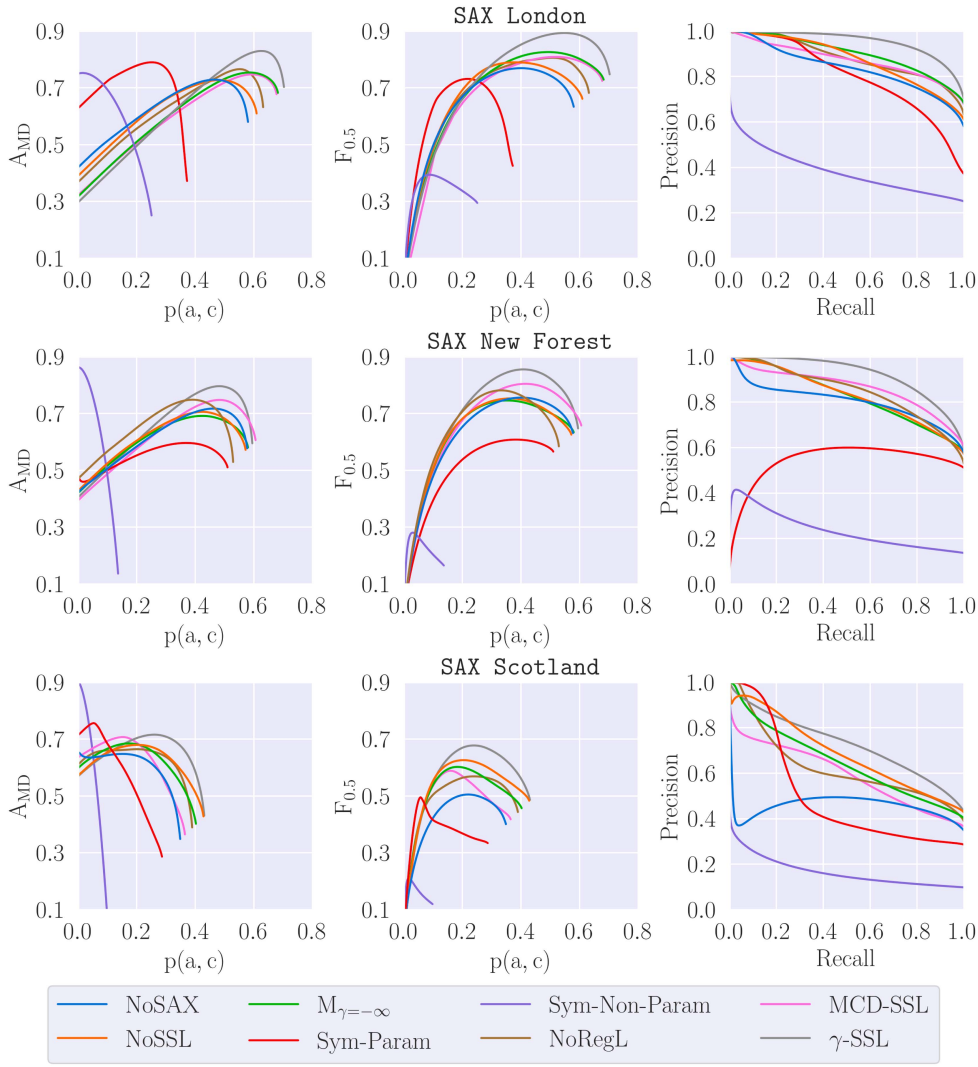


Fig. 9. Misclassification results for ablated γ-SSL models. Given that the ablations are performed on the γ-SSL models, the γ-SSL$_{\text{iL}}$ models are not plotted. See Section IX for details. Best viewed in color.

unit-hypersphere, distance to prototypes is a better measure of uncertainty, and the classes in the target domain become more separable.

F. Is Data Augmentation the Best Way of Obtaining a Distribution Over Possible Segmentations?

Section II discusses how distributional uncertainty can be treated as inherent to the data rather than the model, thus motivating using data augmentation rather than model perturbation to obtain a distribution over possible segmentations of an image. This ablation, MCD-SSL, investigates training the model using dropout instead of data augmentation to calculate M_c. The dropout probability used was 0.2, as this performed best for both misclassification-detection and segmentation performance for the $\text{MCD}_{0.2}$ benchmarks.

MCD-SSL does not achieve as good misclassification-detection performance as γ-SSL, as demonstrated on every metric in Table XIII. This provides evidence that data augmentation is a good method for obtaining segmentation distributions representing the likelihood of correct class assignments.

X. Additional Ablation Studies

In this section differing training procedures are experimented with in order to investigate their performance relative to our proposed method. The experiments involve using the Cityscapes dataset as the labeled dataset, and SAX London data as the unlabeled dataset, with test results reported on the SAX London Test dataset.

A. Does a "soft" M_γ Help Training?

In this experiment, the certainty mask is no longer binary, but instead the confidence is expressed as the max softmax score, $M_\gamma^{(i)} = \text{norm}(\max[\sigma_\tau(s^{(i)})]) \in \{0, 1\}$, where norm is a function that normalizes a batch of certainty masks such that the

lowest pixel confidence is 0, and the highest is 1; see Section IV-A for more details.

The uncertainty estimation results for this are $\mathrm{MaxF}_{0.5}$ is 0.862 at a $\mathrm{p(a, c)}$ of 0.421, and with a segmentation accuracy of 0.576. The results for an equivalent model using a binary thresholded M_γ are as follows: 0.893 @ 0.548 for $\mathrm{MaxF}_{0.5}$ @ $\mathrm{p(a, c)}$ and segmentation accuracy of 0.70, thereby showing a significant drop in segmentation accuracy in the target domain, and also a drop in uncertainty estimation performance.

These results suggest that the soft M_γ introduces noise into the consistency task on the unlabeled domain, and prevents the learning of a high-quality representation of this domain.

B. Should Each Class Prototype Have a Different Threshold?

This experiment evaluates whether learned uncertainty estimation can be improved by thresholding cosine distance between a class prototype and a feature with a different value for each class.

The rationale for this is that datasets often represent different classes with different levels of occurrence and diversity, therefore, the statistics of the each class's representation may also vary, e.g., more diversely represented classes may have a greater variance.

In order to account for this in this experiment, the certainty mask M_γ is calculated from per-class thresholds $\Gamma = [\gamma_1, \gamma_2, \ldots \gamma_K]$ as follows:

$$M_\gamma^{(i)} = \mathbb{1}[\max(s^{(i)}) > \gamma_{k=\mathrm{argmax}(s^{(i)})}] \tag{12}$$

where $s^{(i)} \in \mathbb{R}^K$ are the segmentation scores for a pixel i.

The per-class thresholds Γ are calculated such that the per-class consistency $[p_c]_k$ is equal to the per-class certainty $[p_\gamma]_k$, with both calculated as follows:

$$[p_\gamma]_k = \sum_i^{NHW} \frac{\mathbb{1}[\mathrm{argmax}(s^{(i)}) = k] \odot M_\gamma^{(i)}}{\sum_j^{NHW} \mathbb{1}[\mathrm{argmax}(s^{(j)}) = k]} \tag{13}$$

$$[p_c]_k = \sum_i^{NHW} \frac{\mathbb{1}[\mathrm{argmax}(s^{(i)}) = k] \odot M_c^{(i)}}{\sum_j^{NHW} \mathbb{1}[\mathrm{argmax}(s^{(j)}) = k]}. \tag{14}$$

The uncertainty estimation results achieved with a model trained in this way are significantly worse than our proposed method, with a $\mathrm{MaxF}_{0.5}$ @ $\mathrm{p(a, c)}$ of 0.772 @ 0.432 (versus 0.893 @ 0.548 for our proposed model) and does this with a segmentation accuracy of 0.651 versus 0.703. This suggests that solving for a per-class threshold during training negatively affects both segmentation quality and uncertainty estimation, and it is, therefore, preferable to solve for a single threshold as per our method.

C. Is a Large Batch Size Required for Calculating the Prototypes During Training?

For efficiency during training, the prototypes are calculated from features extracted from a batch of labeled images, whereas in testing, prototypes can be calculated from the entire dataset. This, therefore, raises the question as to whether the training prototypes are too noisy if the batch size becomes too small, and whether a large amount of GPU memory is required for this method. Our proposed training procedure only uses a batch size of 12, and mitigates one aspect of this problem by using the most recent class prototype if a class is not present in the batch of labeled images (shown in Table XIV as use history?).

TABLE XIV
RESULTS FOR VARYING TRAINING PROTOTYPE BATCH SIZE ON SAX LONDON

Batch Size	Use history?	$\mathrm{MaxF}_{0.5}$ @ $\mathrm{p(a, c)}$	Seg. Acc.
12	Yes	0.893 @ 0.548	0.703
8	Yes	0.888 @ 0.559	0.719
6	Yes	0.892 @ 0.546	0.712
2	Yes	0.882 @ 0.518	0.693
2	No	0.827 @ 0.538	0.690

In order to investigate whether small batch sizes are a problem, we train a model with a smaller number of images from which to compute prototypes, while keeping the batch size for the other aspects the same. During testing, prototypes are calculated from all available labeled images from the labeled domain. We report metrics for both the segmentation quality (segmentation accuracy, Seg. Acc.) and uncertainty estimation performance ($\mathrm{MaxF}_{0.5}$ @ $\mathrm{p(a, c)}$) in Table XIV. These results show that, while still using previous prototypes where needed, reducing the prototype batch size *does not* significantly affect segmentation or uncertainty estimation quality. However when not using the history of prototypes, uncertainty estimation quality (in the form of $\mathrm{MaxF}_{0.5}$ @ $\mathrm{p(a, c)}$) does degrade at lower batch sizes, thereby demonstrating the usefulness of this method.

XI. CONCLUSION

Firstly, this article presented a segmentation network that mitigates misclassification on challenging distributionally shifted test data via uncertainty estimation. It achieved this by learning a feature representation, where pixel embeddings corresponding to accurate and inaccurate segmentations are separable by a single global threshold around prototypical class features. By leveraging a large quantity of uncurated unlabeled data from the deployment domain, the constraint of having labeled data from that domain is relaxed, and thus, a small labeled dataset from a distinct domain can be used. Second, it presented a novel semantic segmentation test benchmark, comprising a set of 700 pixel-wise labels from three distinct domains and metrics to measure quality of uncertainty estimation. Upon evaluation on this challenging benchmark, the presented network outperformed epistemic uncertainty estimation and OoD detection methods, and does so without increasing the computational footprint of a standard segmentation network.

REFERENCES

[1] C. Galindo, J.-A. Fernández-Madrigal, J. González, and A. Saffiotti, "Robot task planning using semantic maps," *Robot. Auton. Syst.*, vol. 56, no. 11 , pp. 955–966, 2008.

[2] G. Pramatarov, D. De Martini, M. Gadd, and P. Newman, "Box-graph: Semantic place recognition and pose estimation from 3D LiDAR," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 7004–7011.

[3] I. Kostavelis and A. Gasteratos, "Semantic mapping for mobile robotics tasks: A survey," *Robot. Auton. Syst.*, vol. 66, pp. 86–103, 2015.

[4] M. Gadd, D. De Martini, L. Marchegiani, P. Newman, and L. Kunze, "Sense–Assess-eXplain (SAX): Building trust in autonomous vehicles in challenging real-world driving scenarios," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2020, pp. 150–155.

[5] *Assuring the Operational Safety of Automated Vehicles–Specification*, British Standards Institution, Standard PAS 1881, 2022.

[6] *Road Vehicles-Safety of the Intended Functionality*, International Organization for Standardization, Standard ISO 21448, 2019.

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

[8] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017.

[9] C. Gurau, A. Bewley, and I. Posner, "Dropout distillation for efficiently estimating model confidence," 2018, *arXiv:1809.10562*.

[10] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *Proc. Int. Conf. Learn. Representations*, 2018.

[11] J. Liu, Z. Lin, S. Padhy, D. Tran, T. B. Weiss, and B. Lakshminarayanan, "Simple and principled uncertainty estimation with deterministic deep learning via distance awareness," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 7498–7512.

[12] J. Mukhoti, A. Kirsch, J. van Amersfoort, P. H. Torr, and Y. Gal, "Deterministic neural networks with inductive biases capture epistemic and aleatoric uncertainty," 2021, *arXiv:2102.11582*.

[13] J. R. van Amersfoort, L. Smith, Y. W. Teh, and Y. Gal, "Simple and scalable epistemic uncertainty estimation using a single deep deterministic neural network," in *Proc. Int. Conf. Mach. Learn.*, 2020.

[14] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017.

[15] R. Weston, S. H. Cen, P. Newman, and I. Posner, "Probably unknown: Deep inverse sensor modelling radar," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 5446–5452.

[16] J. Gast and S. Roth, "Lightweight probabilistic deep networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3369–3378.

[17] D. Novotny, S. Albanie, D. Larlus, and A. Vedaldi, "Self-supervised learning of geometrically stable features through probabilistic introspection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 3637–3645.

[18] D. Hendrycks and K. Gimpel, "A baseline for detecting misclassified and out-of-distribution examples in neural networks," in *Proc. Int. Conf. Learn. Representations*, 2017.

[19] S. Liang, Y. Li, and R. Srikant, "Enhancing the reliability of out-of-distribution image detection in neural networks," in *Proc. Int. Conf. Learn. Representations*, 2018.

[20] K. Lee, K. Lee, H. Lee, and J. Shin, "A simple unified framework for detecting out-of-distribution samples and adversarial attacks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018.

[21] H. Wang, Z. Li, L. Feng, and W. Zhang, "ViM: Out-of-distribution with virtual-logit matching," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 4921–4930.

[22] I. Golan and R. El-Yaniv, "Deep anomaly detection using geometric transformations," in *Proc. Adv. neural Inf. Process. Syst.*, 2018.

[23] D. Hendrycks, M. Mazeika, S. Kadavath, and D. Song, "Using self-supervised learning can improve model robustness and uncertainty," *Adv. neural Inf. Process. Syst.*, 2019.

[24] D. Hendrycks, M. Mazeika, and T. G. Dietterich, "Deep anomaly detection with outlier exposure," in *Proc. Int. Conf. Learn. Representations*, 2019.

[25] A. Malinin and M. Gales, "Predictive uncertainty estimation via prior networks," in *Proc. Adv. neural Inf. Process. Syst.*, 2018.

[26] K. Lee, H. Lee, K. Lee, and J. Shin, "Training confidence-calibrated classifiers for detecting out-of-distribution samples," in *Proc. Int. Conf. Learn. Representations*, 2018.

[27] D. S. W. Williams, M. Gadd, D. D. Martini, and P. Newman, "Fool me once: Robust selective segmentation via out-of-distribution detection with contrastive learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 9536–9542.

[28] A. Tarvainen and H. Valpola, "Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017.

[29] S. Laine and T. Aila, "Temporal ensembling for semi-supervised learning," in *Proc. Int. Conf. Learn. Representations*, 2017.

[30] Q. Xie, Z. Dai, E. Hovy, T. Luong, and Q. Le, "Unsupervised data augmentation for consistency training," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 6256–6268.

[31] M. Assran et al., "Semi-supervised learning of visual features by non-parametrically predicting view assignments with support samples," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 8443–8452.

[32] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 9912–9924.

[33] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. neural Inf. Process. Syst.*, 2017.

[34] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016.

[35] G. Kang, L. Jiang, Y. Yang, and A. G. Hauptmann, "Contrastive adaptation network for unsupervised domain adaptation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 4893–4902.

[36] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial discriminative domain adaptation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 7167–7176.

[37] K. Wong, S. Wang, M. Ren, M. Liang, and R. Urtasun, "Identifying unknown instances for autonomous driving," in *Proc. Conf. Robot Learn.*, 2020, pp. 384–393.

[38] M. Siam, A. Kendall, and M. Jagersand, "Video class agnostic segmentation benchmark for autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2021, pp. 2825–2834.

[39] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 1597–1607.

[40] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 9729–9738.

[41] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9929–9939.

[42] P. Mettes, E. van der Pol, and C. G. M. Snoek, "Hyperspherical prototype networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019.

[43] J.-B. Grill et al., "Bootstrap your own latent—A new approach to self-supervised learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 21271–21284.

[44] M. Cordts et al., "The cityscapes dataset for semantic urban scene understanding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3213–3223.

[45] F. Yu et al., "Bdd100 k: A diverse driving dataset for heterogeneous multitask learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2636–2645.

[46] H. Alhaija, S. Mustikovela, L. Mescheder, A. Geiger, and C. Rother, "Augmented reality meets computer vision: Efficient data generation for urban driving scenes," in *Proc. Int. J. Comput. Vis.*, 2018, pp. 961–972.

[47] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 801–818.

[48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.

[49] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 41–48.

[50] Y. Gal, "Uncertainty in deep learning," Doctoral dissertation, Dept. of Eng., Univ. of Cambridge, Cambridge, U.K., 2016.

[51] J. Mukhoti and Y. Gal, "Evaluating Bayesian deep learning methods for semantic segmentation," 2018, *arXiv:1811.12709*.

[52] J. Behrmann, W. Grathwohl, R. T. Chen, D. Duvenaud, and J.-H. Jacobsen, "Invertible residual networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 573–582.

[53] Y. Ovadia et al., "Can you trust your model's uncertainty? Evaluating predictive uncertainty under dataset shift," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019.

[54] O. Zendel, K. Honauer, M. Murschitz, D. Steininger, and G. F. Dominguez, "Wilddash—Creating hazard-aware benchmarks," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 402–416.