U-SEG: Uncertainty in SEGmentation - A systematic multi-variable exploration

Supplementary Material

A1. Implementation Details

A1.1. Datasets

As discussed in Sec. 4, we evaluate over the VIPER [16] and Cityscapes [2] datasets, both of which are driving-focused. This somewhat limited selection is due to the requirements we have for our datasets. First, they must have a minimum level of complexity, as we are evaluating uncertainty estimation approaches whose primary goal is to assist with real world issues. Second, we wish to be able to evaluate distribution shifts, which requires distinct datasets that have significant annotation overlap. Third, in order to evaluate out-of-distribution detection we need to be able to acquire appropriate out-of-distribution images. Fourth, the time series prediction model requires prior frames to exist and to be close enough (in time/space) to the annotated frames such that we can align them via optical flow or some other method. Lastly, the dataset needs to have panoptic annotations or, as in our case, semantic and instance annotations that can be converted to panoptic.

Both datasets are high resolution, with images being a minimum of 1024 pixels in the smallest dimension before any processing. The Cityscapes and VIPER training set include 2975 and 13367 images respectively, with their validation sets including 500 and 4959. When it comes to time series data, Cityscapes provides un-annotated frames for an interval before and after the annotated frames we evaluate on, while in VIPER all frames are annotated but we choose to only use primary set consisting of an annotation for every 10 frames so as to effectively match the Cityscapes configuration. During evaluation we only use at most 5 prior frames.

We note that one of the limitations of our study is the exclusive use of driving related datasets. We emphasize that this is not a requirement of our proposed approach or our study. We believe that our findings generalize to other domains, but to our knowledge there are no non-driving datasets that match our requirements for us to evaluate on.

With respect to the distribution shift variants of each dataset, please see Sec. A1.3 and Tabs. A1 and A2.

A1.2. Optical Flow

For the VIPER dataset, ground truth optical flow provided by the authors is used. This data is high quality as the VIPER dataset is artificially generated, but there are some gaps in the data where an "interruption" occurs (consecutive frames in a sequence have a much larger gap than normal and lack the relevant flow data). There are also parts of any given image where there is simply no flow data *e.g*. any pixels with a pedestrian.

For the Cityscapes dataset, we use the popular RAFT [19] to generate the optical flow. We use the `torchvision` implementation and the `raft_large` variant with weights `DEFAULT` as we found them to work best. Default model parameters are used, with images resized to $\frac{3}{4}$ normal size (*i.e*. 1536×768) due to computational constraints.

In both datasets, there are many instances of gaps where there are no prior time series frames. Beyond some missing frames in the VIPER dataset, these are primarily due to the start of a new driving sequence. In such cases, we do not use any prior time series frames, meaning that no prior frames can be sampled if the time series prediction model is in use for those frames. Such occurrences are rare however and should not meaningfully affect our results.

A1.3. Distribution Shifts

In Tabs. A1 and A2 we provide the mapping of classes from VIPER [16] to Cityscapes [2] and vice-versa. We use these in the distribution shift scenario where we evaluate a model trained on one dataset on the other. While both datasets are focused on driving, and thus have significant overlap in definitions, it is not always one-to-one. For example, the VIPER `infrastructure` class represents a number of objects such as walls, overpasses and street lights. However, we are forced to map it to Cityscapes `VOID` as there is no equivalent and no way to split out the matching objects such as walls. Going the other way, however, we can map `wall` in Cityscapes to `infrastructure` in VIPER. We further note that while most classes are treated the same in panoptic segmentation, being semantic only labels or including instance annotations, there are some differences such as `traffic light`. Going from a model trained to expect that class as one requiring individual instance annotation to evaluation that does not should not cause any issues. The opposite direction, however, would be more problematic and prone to errors where the instance IDs are not properly assigned. The general design of the network where potential objects are predicted (Eq. (3)) means that an error is not guaranteed, though, and correct instance labels are still possible.

A1.4. Time series

Our implementation of using time series frames in an ensemble is primarily limited by errors in the alignment of prior frames to the current one. In the case of Cityscapes, we have any errors made by RAFT, while for VIPER even the ground truth remains limited by standard optical flow constraints such as occlusions. The effect of these errors can be seen

VIPER			→	Cityscapes		
ID	**Class Name**	**Instances**		**ID**	**Class Name**	**Instances**
13	trafficlight	✓		19	traffic light	✗
16	firehydrant	✓		0	VOID	✗
17	chair	✓		0	VOID	✗
19	trashcan	✓		0	VOID	✗
20	person	✓		24	person	✓
23	motorcycle	✓		32	motorcycle	✓
24	car	✓		26	car	✓
25	van	✓		26	car	✓
26	bus	✓		28	bus	✓
27	truck	✓		27	truck	✓
2	sky	✗		23	sky	✗
3	road	✗		7	road	✗
4	sidewalk	✗		8	sidewalk	✗
6	terrain	✗		22	terrain	✗
7	tree	✗		21	vegetation	✗
8	vegetation	✗		21	vegetation	✗
9	building	✗		11	building	✗
10	infrastructure	✗		0	VOID	✗
11	fence	✗		13	fence	✗
12	billboard	✗		0	VOID	✗
14	trafficsign	✗		20	traffic sign	✗
15	mobilebarrier	✗		0	VOID	✗
18	trash	✗		0	VOID	✗
0	VOID	✗		0	VOID	✗

Table A1. Mapping of labels from VIPER to Cityscapes. VOID denotes the background/ignore class; the Instances column indicates whether instance annotations are provided in the corresponding dataset. ID numbers follow the respective dataset definitions as used in ground truth. Following standard practice only classes that are normally evaluated (*i.e.* with a valid contiguous training ID) are considered.

in Sec. A7.3 and most notably in Fig. A3 where panoptic segmentation, which is more sensitive to any errors given the problem definition, is hampered more than semantic segmentation as the number of prior frames increases. [6] propose a second method that relies on reconstruction error to address this, but we elected to not implement it as we did not wish to add further (potentially confounding) variables to our study.

A1.5. Foundation Model Performance

As mentioned in Secs. 3.2 and 4, we were inspired by the performance of foundation models like DINOv2 [14] on the BRAVO challenge, where they demonstrated excellent performance on a range of challenging tasks including out-of-distribution detection, calibration or simply handling adverse weather conditions. As we explain in Sec. 5, however, we do not see the same dominance with another backbone performing the same or better in many cases. For failure detection in particular, which is very sensitive to segmentation performance, our inability to train a DINOv2 backbone to match or exceed a Swin-B or ResNet50 alternative is likely a notable contributing factor.

In Tab. A3, we compare our trained DINOv2 models against a ResNet50 on panoptic and semantic segmentation performance. We see that for the Cityscapes and VIPER datasets, DINOv2 performance lags a basic ResNet50 (with the exception of mIoU for VIPER). However, on the more diverse ADE20K dataset, a smaller DINOv2 variant with the ViT-S architecture is easily able to beat a ResNet50. We made many attempts to improve DINOv2 performance, including different transfer learning sources, longer training, and freezing the DINOv2 model except the *qkv* and/or projections layers. All were unsuccessful, and we ultimately conclude that the lack of diversity in Citycapes and VIPER is the primary cause given the ADE20K performance.

A1.6. Training parameters

Where possible, trained models from [1] were used, covering ResNet50 and Swin-B on Cityscapes. All other models were trained by us. We attempted to follow as best as possible the training parameters used by [1], including the number of epochs (steps), preprocessing settings, data augmentations, optimizer settings, learning rates, and other

Cityscapes			→	VIPER		
ID	**Class Name**	**Instances**		**ID**	**Class Name**	**Instances**
7	road	✗		3	road	✗
8	sidewalk	✗		4	sidewalk	✗
11	building	✗		9	building	✗
12	wall	✗		10	infrastructure	✗
13	fence	✗		11	fence	✗
17	pole	✗		10	infrastructure	✗
19	traffic light	✗		13	trafficlight	✓
20	traffic sign	✗		14	trafficsign	✗
21	vegetation	✗		8	vegetation	✗
22	terrain	✗		6	terrain	✗
23	sky	✗		2	sky	✗
24	person	✓		20	person	✓
25	rider	✓		20	person	✓
26	car	✓		24	car	✓
27	truck	✓		27	truck	✓
28	bus	✓		26	bus	✓
31	train	✓		0	VOID	✗
32	motorcycle	✓		23	motorcycle	✓
33	bicycle	✓		0	VOID	✗
0	VOID	✗		0	VOID	✗

Table A2. Mapping of labels from Cityscapes to VIPER. VOID denotes the background/ignore class; the Instances column indicates whether instance annotations are provided in the corresponding dataset. ID numbers follow the respective dataset definitions as used in ground truth. Following standard practice only classes that are normally evaluated (*i.e.* with a valid contiguous training ID) are considered.

Backbone	**Dataset**	**PQ**	**mIoU**
DINOv2 (ViT-B)	Cityscapes	55.73	75.18
ResNet50†	Cityscapes	**62.09**	**77.41**
DINOv2 (ViT-S)	ADE20K	**42.99**	**50.06**
ResNet50†	ADE20K	39.51	45.81
DINOv2 (ViT-B)	VIPER	42.72	**73.76**
ResNet50	VIPER	**43.39**	69.07

Table A3. Segmentation performance comparison of different backbones across datasets using a baseline (deterministic) Mask2Former model. All runs use a fixed seed of 1. † indicates model provided by [1]; training is done by us otherwise.

model parameters. For ResNet50 and Swin-B on VIPER, transfer learning is used from corresponding COCO [10] trained models. For DINOv2 on VIPER, the Mask2Former head is initialized with weights from a COCO-trained Swin-B model, while on Cityscapes it is randomly initialized. Transfer learned models are sourced from [1]. The DINOv2 backbone itself is initialized with weights from [14] from the ViT-B/14 distilled with registers model for both datasets.

A1.7. Calibration

For calibration, we use 15 bins following [5, 20]; most literature appears to use a value between 10 and 20. As well, following [11] we implement argmax calibration, where only the calibration of the highest likelihood score is evaluated.

A2. Prediction Models

In Tab. A4, we show all the prediction model configurations we test. Most experiments were evaluated with all of those shown with the Mask Distance sample aggregation (**V4**) method. The other two approaches are used only in Sec. A7.4 for the comparison of different approaches. Fewer configurations were evaluated for the aggregation comparisons compared to the main experiments due to implementation limits (*e.g.* the pixel decoder is incompatible with the scale TTA transform), resource constraints or computational limits. For example, the Averaging [18] approach has GPU memory requirements linear with the number of samples; at higher levels, this requires very large GPUs. The Mask Distance approach in contrast has constant GPU memory requirements with respect to the number of samples.

Other prediction model parameters are as follows:

- **MC Dropout**:
  - Dropout rate fixed at $p = 0.1$.

Prediction Model Configuration				Applicable Sample Aggregation (V4)		
MC Drop.	**Prev. Time Frames**	**TTA Transf.**	**Total # Samples**	**Mask Dist.**	**Pixel Dec.**	**Averaging[18]**
0	0	None	1	-	-	-
0	0	Horizontal Flip	2	✓	✓	✓
0	0	Scale + Horizontal Flip	6	✓	✗	✓
0	0	Scale	3	✓	✗	✓
0	1	None	2	✓	✓	✓
0	1	Horizontal Flip	4	✓	✓	✓
0	1	Scale + Horizontal Flip	12	✓	✗	✓
0	1	Scale	6	✓	✗	✓
0	2	None	3	✓	✓	✓
0	2	Horizontal Flip	6	✓	✓	✓
0	2	Scale + Horizontal Flip	18	✓	✗	✓
0	2	Scale	9	✓	✗	✓
0	3	None	4	✓	✗	✗
0	3	Horizontal Flip	8	✓	✗	✗
0	3	Scale	12	✓	✗	✗
0	5	None	6	✓	✗	✗
0	5	Horizontal Flip	12	✓	✗	✗
0	5	Scale	18	✓	✗	✗
3	0	None	3	✓	✓	✓
3	0	Horizontal Flip	6	✓	✗	✗
3	0	Scale + Horizontal Flip	18	✓	✗	✗
3	0	Scale	9	✓	✗	✗
3	1	None	6	✓	✓	✓
3	1	Horizontal Flip	12	✓	✓	✓
3	1	Scale	18	✓	✗	✓
3	2	None	9	✓	✗	✗
3	2	Horizontal Flip	18	✓	✗	✗
3	3	None	12	✓	✗	✗
3	5	None	18	✓	✗	✗
5	0	None	5	✓	✗	✗
5	0	Horizontal Flip	10	✓	✗	✗
5	0	Scale	15	✓	✗	✗
5	1	None	10	✓	✗	✗
5	1	Horizontal Flip	20	✓	✗	✗
5	2	None	15	✓	✗	✗
5	3	None	20	✓	✗	✗
10	0	None	10	✓	✗	✗
10	0	Horizontal Flip	20	✓	✗	✗
10	1	None	20	✓	✗	✗

Table A4. List of all prediction model configurations tested, alongside the applicable Sample Aggregation approach under which they were tested. All 39 configurations were tested with Mask Distance for most experiments. The first row is the baseline approach. Approaches other than Mask Distance were only used for the comparison of sample aggregation approaches in Sec. A7.4.

  - Dropout layers are only activated when MC Dropout is enabled. Thus, all non-MC Dropout configurations are deterministic.
  - The only Dropout layers activated in the network are those in the head; specifically, in the pixel decoder. Following [1], we use the multi-scale deformable attention Transformer (MSDeformAttn) [21] as the pixel decoder.
- **TTA**: The following transforms are used in TTA:
  - Horizontal flips, where the image is flipped horizontally. Flips were shown to perform well in [8].
  - Scale transformations: These result in two distinct samples from rescaling operations; the first at 0.8x and the

second at 1.25x. Scale values were chosen following [4].

Of note, unlike [8] we do not use Gaussian noise as we observed significant performance degradation. We believe this to be a result of the fact that such noise is not part of the normal augmentations applied during training.

A3. Sample Aggregation Details

In this study, we introduce Mask Distance to address the problem of sample aggregation. This problem exists because of our choice of a universal architecture model (Mask2Former), which was in turn chosen as a result of our desire to cover both the panoptic and semantic segmentation domains. Complementary work such as [8] does not require such a step as the task of semantic segmentation produces a class distribution at each pixel, and aggregating them is as straightforward as taking the mean. With instances, however, we must first determine the correspondences.

Other work on uncertainty and panoptic segmentation such as [17] do not use ensembles and thus sample aggregation is not a problem. Were ensembles to be validated on such network designs, some form of sample aggregation would be needed; given the design of the panoptic fusion model in [17] for instance this is likely to be a complex process.

For a comparison of the various sample aggregation approaches, please see Sec. A7.4 and Fig. A9.

A3.1. Mask Distance

As presented in Sec. 4.3, the key detail of the Mask Distance approach is that the relatively straightforward computation of the Euclidean distance between all the masks representing potential objects in each sample is sufficient to determine the correspondence without any consideration of classes. As presented, however, Eq. (4) obfuscates a notable implementation detail: we calculate the Euclidean distance between the mask of one sample against the next, but only for the first two samples. Subsequent samples are then matched against a running average of the samples accumulated up until that point. This eliminates the need to keep more samples in limited GPU memory, as well as the potential for one or more poor quality samples to lead to bad or missing matches. The memory requirements for Mask Distance are constant with respect to the number of samples. This allows us to evaluate prediction models combinations with up to 20 samples on a single 12GB GPU.

A3.2. Pixel Decoder

With the pixel decoder, we average the samples from the ensemble inside the segmentation head. Specifically, we calculate the average at the four outputs of the pixel decoder in Fig. 2 of [1]. We then execute the remainder of the head as normal, giving us a single prediction. Not having access to samples at the output of the network however limits us in terms of the variety of uncertainty measures we can calculate, as shown in Tab. A5. Attempting to merge intermediate network features also places some restrictions on applicable prediction models such as any TTA configuration with multiple scales, shown in Tab. A4. As this approach performs poorly in terms of uncertainty and segmentation performance, shown in Sec. A3 and Fig. A9, we did not investigate this approach further in favour of Mask Distance. It does, however, do well in terms of the time needed to aggregate samples, as shown in Sec. A8.

A4. Uncertainty Measures

In Tab. A5, we list all the uncertainty measures we calculate and indicate which sample aggregation methods (**V4**) they can be calculated with. We include both typical well-defined measures of uncertainty such as the Predictive Entropy (Eq. (2)) as well as more ad-hoc versions such as the variance. As the network produces two outputs (Eq. (3)), we also ensure that we apply a variety of measures that are able to fully capture all information contained in said outputs.

This results in two different general approaches: *Class & Mask* and *Mask*. The former is applied by using the mask assignments as weights for a weighted sum of the softmax output following [1]. The result at each pixel is thus a class distribution whose composition has been influenced by potential assignments to one or more objects. For the latter approach, we ignore the class output and use only the mask output with measures such as the Expected Mask Variance or Mutual Information (Mask). This is inspired from prior work that such as [3, 12] that explicitly calculate a measure of spatial uncertainty. Towards this end, we further introduce the Maximum Normalized Sigmoid Mask Score, which is simply the maximum of the mask output of the network passed through a sigmoid function and normalized to add to 1. This represents the maximum uncertainty the network has with respect to its instance assignments at each pixel. We further introduce the Combined Maximum Softmax & Normalized Sigmoid, which is defined to be the Maximum Softmax Score at all pixels except those determined by the network to require instance annotations (*things* in panoptic segmentation), where it is instead the mean of the Maximum Softmax Score and the Maximum Normalized Sigmoid Mask Score. These are very ad-hoc measures, but we wish to give downstream tasks the most choice possible as we optimize over the uncertainty measure in most experiments.

A5. Optimization

In this work, in many experiments we chose to optimize over variables such as the uncertainty measure (**V6**) or pixel aggregation approach (**V7**). This allows the best performance

Uncertainty Measure	Applicable Sample Aggregation			
	Baseline (None)	**Mask Dist.**	**Pixel Dec.**	**Averaging [18]**
Predictive Entropy (Class & Mask)	✓	✓	✓	✓
Predictive Entropy (Mask)	✓	✓	✓	✗
Expected Entropy (Mask)	✗	✓	✗	✓
Expected Entropy (Class & Mask)	✗	✓	✗	✓
Mutual Information (Mask)	✗	✓	✗	✗
Mutual Information (Class & Mask)	✗	✓	✗	✓
Expected Mask Variance	✗	✓	✗	✓
Predictive Mask Variance	✓	✓	✓	✗
Maximum Softmax Score (Class & Mask)†	✓	✓	✓	✓
Maximum Normalized Sigmoid Mask Score	✓	✓	✓	✗
Combined Maximum Softmax & Normalized Sigmoid	✓	✓	✓	✗

Table A5. Uncertainty measures evaluated in the paper. A checkmark indicates that the measure can be calculated with the corresponding sample aggregation approach (**V4**), while an X indicates that it cannot be calculated. Baseline represents no sample aggregation *i.e*. a deterministic network, where measures are mapped to the nearest approximation. For example, rather than predictive entropy following Eq. (1) we use the softmax entropy [13]. † indicates the measure is used for calibration experiments.

possible for the variables under study, without introducing any bias as a result of fixing the choices as shown by [8]. As an example, assume the best possible result achievable with a deterministic baseline relies on Predictive Entropy, but with MC Dropout where we can calculate Expected Entropy (as we have > 1 sample), we can do better. As shown in Tab. A5, there are a number of incompatible sample aggregation and uncertainty measure combinations. Going back to our example, if we fix the uncertainty measure to Expected Entropy, how could we fairly compare with a baseline? Going the other direction, if we fix the uncertainty measure to Predictive Entropy, we then ignore any performance gains achievable with other uncertainty measures that are not universally compatible, effectively holding them back. As we show in the main paper, in deployment a practitioner will need to make a choice, jointly considering all variables.

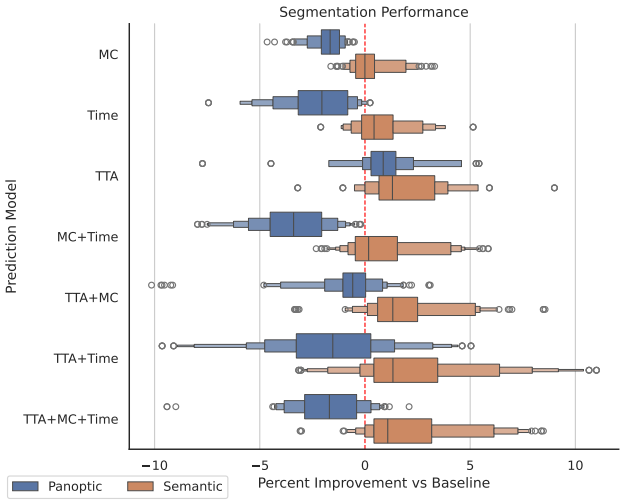
A6. Metrics

For out-of-distribution detection, when calculating the AUROC metric we use the `roc_auc_score` function from scikit-learn [15]. The uncertainty measures are treated as the target score, while the classification of the image as out-of-distribution or not serves as the label. For failure detection, we calculate per-image IoU and PQ metrics for semantic and panoptic segmentation. These per-image metrics are calculated across all classes; we cannot calculate standard metrics where an average is taken across all classes (*e.g*. Sec. 4.2 in [9]) as unlike with an entire dataset there may only be a few classes present in each image. These are then used to calculate the risk and subsequent AURC metric following [7, 8].

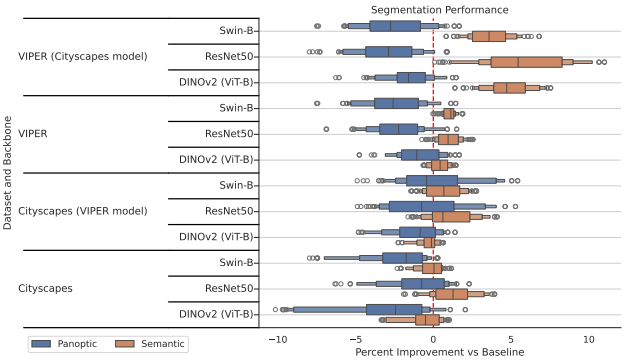
For distribution shifted versions of both datasets, when evaluating segmentation metrics we ignore any classes for which there is no ground truth to evaluate. When evaluating panoptic segmentation, we always report the PQ metric for all categories, outside of per-image evaluation used in failure detection where it is not applicable.

In Sec. 4, we state that we attempt to preserve "objective compatibility" during evaluation. By this, we mean that we choose to use metrics in such a way that we can fairly compare them across semantic and panoptic segmentation. Obviously, direct comparisons are impossible, but relative comparisons as we show in many plots are possible given the right conditions. Specifically, the metrics need to be measuring a quantity directly tied to the domain, such that a change in domain results in an easy to understand change to the metric as a direct result of the domain change and nothing else. For semantic segmentation, we have the mIoU which represents the average across classes of per-pixel label assignment. When we go to panoptic segmentation, this becomes PQ, which also represents the average across classes but now of the correct segment assignment. This extends to failure detection (AURC-PQ vs AURC-IoU) and calibration (ECE). As we state in Sec. 4.4, for calibration we forgo existing proposed metrics such as pECE [17] as a reliance on segment matching in panoptic segmentation has no direct correlation in semantic segmentation. With our implementation, we evaluate the ECE at the pixel level for both domains; the difference lies in what is used to determine if a pixel is correct. In semantic segmentation, it is the class label; in panoptic segmentation, it is both the class label and the instance label as used in the PQ metric calculation.

As shown in Tab. A5, for all calibration experiments we use the Maximum Softmax Score (Class & Mask). We chose this approach due to its simplicity and no potential for confounding variables. [8] by contrast use Platt scaling to be



(a)



(b)

Figure A1. Normalized segmentation performance, by (a) prediction model and (b) dataset and backbone. Segmentation performance is not a downstream task but nonetheless is important being a key part of the problem definition. Use of the letter-value plot and baseline definition follow Fig. 2. Variables **V1** to **V3** and **V5** are swept through and shown, with **V4** fixed to the best performer *Mask Distance*. Variables **V6** and **V7** are not applicable for this task.

able to evaluate multiple sample aggregation measures, but this requires a held out dataset (which we do not have) and has the potential to add bias from the learned Platt scaling parameters. They also use ACE (Average Calibration Error) as their metric of choice due to the predominance of background annotated pixels in their experiments. As our datasets are not from the medical imaging domain, we do not experience this issue and thus the ECE is appropriate.

A7. Additional Results

A7.1. Segmentation Performance

In Fig. A1, we compare the normalized segmentation performance across various prediction models, datasets, and

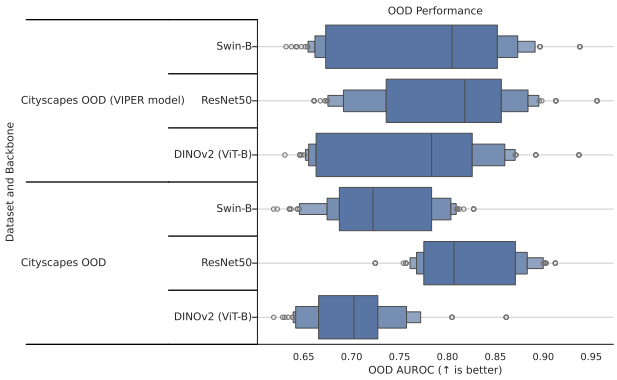


Figure A2. Out-of-distribution performance by dataset (**V1**) and backbone (**V2**) without normalization using the Area Under the Receiver Operating Characteristic metric. Experimental procedure follows Fig. 3.

backbones. We see that the relative performance between the various categories roughly match that seen in the failure detection plots (Fig. 2). Thus, while the objective of any uncertainty estimation approach is to generate a high quality uncertainty estimate, any unintended effects on segmentation performance can be a significant contributor to (potentially poor) performance on some downstream tasks for which uncertainty estimates are used such as failure detection. This calls into question the claims made by some that "not overly compromising performance on the base task" [3] in exchange for better uncertainty estimates is acceptable, at least for downstream tasks such as failure detection.

A7.2. Out-of-distribution performance

In Fig. A2, we show the OOD performance without normalization on the Cityscapes dataset and with a distribution shift where we use the VIPER model instead. We see that un-normalized results are broadly in line with Figs. 3 and 4, with a fairly broad spread of results and no clear performance leader, including for the foundation DINOv2 model which tends to trail other backbones in absolute terms.

A7.3. Downstream Tasks

In Figs. A3 and A4, we show failure detection and calibration results respectively from the same experiments as Fig. 2a but broken down further into the individual prediction model configurations we evaluate. Most of the relevant observations are discussed in Sec. 5, but we do note the fact that time series frames perform worse the more frames are added in the failure detection case. This applies to both semantic and panoptic segmentation, but is more pronounced in the latter case. Meanwhile, for panoptic calibration we do see some degradation with increased number of time series frames but it is fairly minimal.

In Fig. A5, we break down the results shown in Fig. 3a into the individual prediction model configurations. We

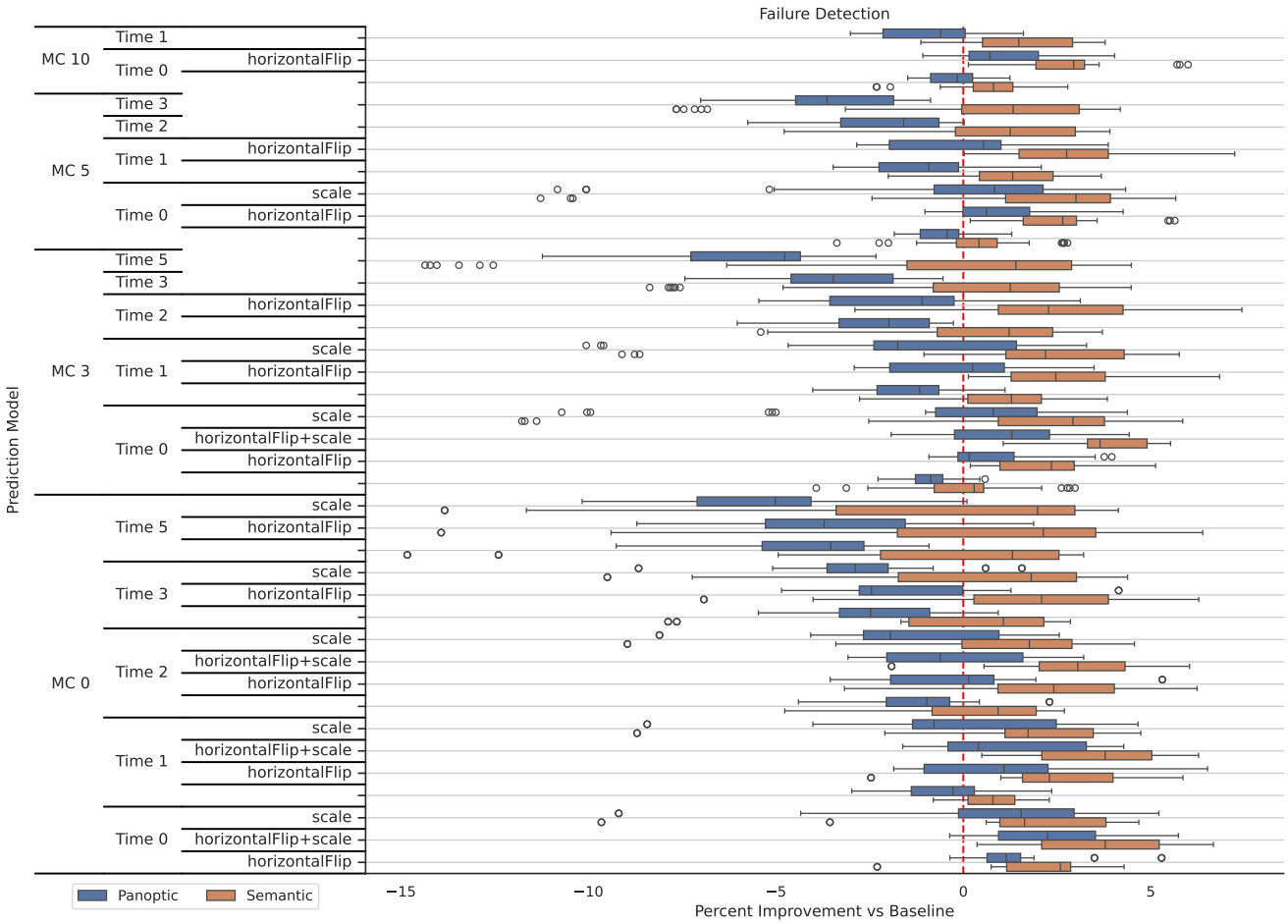


Figure A3. Results on the downstream task (**V8**) of failure detection, plotted by prediction model (**V3**) and broken down into each individual tested configuration. Experiment parameters follow Fig. 2a, except a traditional box plot is used.

see that despite the high performance of time series data as explained in Sec. 5, performance degrades with more samples. Part of this may be due to noise from the frame alignment process (optical flow errors, occlusions) but it is also likely an artifact of the implementation. The out-of-distribution task expect higher uncertainty of OOD scenarios, but using more images without the OOD object from prior frames will decrease overall uncertainty explaining the worse results.

In Figs. A6 to A8 we extend the results shown in Figs. 2a and 3a by additionally breaking the results down by the number of samples (see Tab. A4). We see the same trends previously discussed in this section, where using more time series frames is detrimental in some cases due to noise but less so in others.

Comparing across downstream tasks in Figs. A3 to A5, the relative performance differences are a result of the task specification and the underlying properties of the evaluation metrics. For instance, in Fig. A3 we see that adding more time series frames hurts performance, but in Fig. A4 the effect exists but is minimal in comparison. This is not surprising, as the increased noise (previously discussed in this section) from adding more frames harms segmentation performance more than any increase in failure detection from a better uncertainty estimate. Meanwhile, calibration error is independent of segmentation error, although the noise from alignment errors as more frames are added still causes a (minimal) decrease.

A7.4. Sample Aggregation

In Fig. A9, we see that our choice to use *Mask Distance* in all other experiments is justified, as it leads across the board on segmentation performance and the downstream tasks of calibration and failure detection. This applies for both panoptic and semantic segmentation. Please see Secs. A3 and 4 for details.

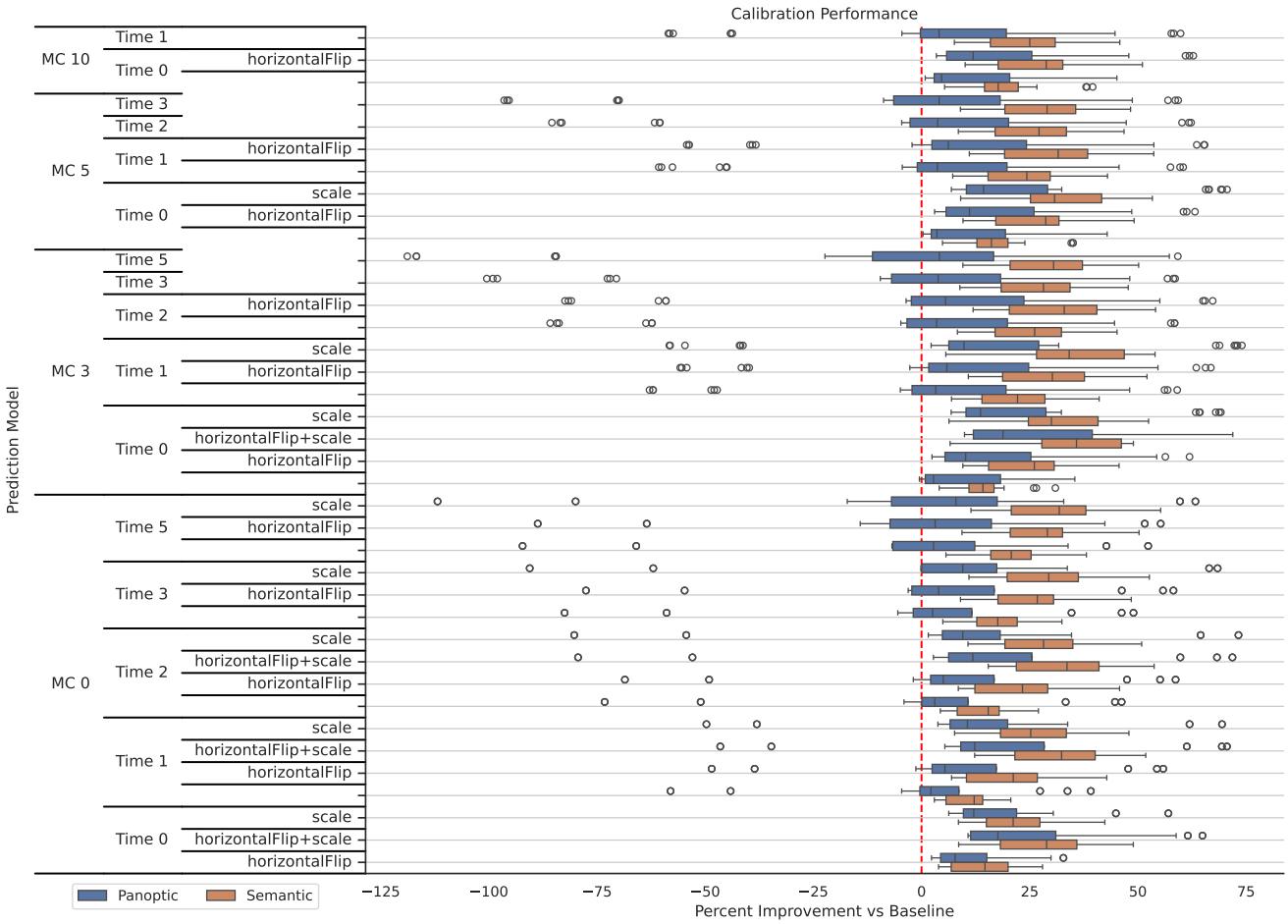


Figure A4. Results on the downstream task (**V8**) of calibration, plotted by prediction model (**V3**) and broken down into each individual tested configuration. Experiment parameters follow Fig. 2a, except a traditional box plot is used.

A7.5. Uncertainty Measures

In Figs. A10 and A11, using the data from our main experiments as presented in Figs. 2 and 3 we compare different uncertainty measures and pixel aggregation approaches. Note that this is a specific case where we wished to examine if one uncertainty measure or pixel aggregation was dominant; in all other experiments, we optimize over uncertainty measures and pixel aggregation approaches so as to avoid introducing any bias as discussed in Sec. A4.

What the results show is broadly in line with [8] in that the best possible approach to take is very conditional, with a joint optimization of all variables being necessary when choosing the best setting for each variable. Were we forced to make an overall choice, image level aggregation seems the best, but there is no clear winner for uncertainty measures. In failure detection, predictive entropy leads in the panoptic domain, while the ad-hoc measure that combines the softmax and normalized mask score is roughly tied with expected entropy in the semantic domain. Meanwhile, in out-of-distribution detection mutual information is a clear leader.

A7.6. Domain Changes

In Figs. 2 to A4, we compare results across both the panoptic and semantic segmentation domains. In the main paper, we note that panoptic performance is usually worse than semantic on a given metric. We believe the primary reason for this is the problem specification, where semantic segmentation requires a class label and panoptic segmentation requires the both a class and instance label for each pixel. The metrics used for each (discussed in Sec. 4.4) are also different, with panoptic segmentation requiring individual objects to achieve a minimum overlap with ground truth. Meanwhile, semantic segmentation only evaluates pixel-level correctness for each class. This makes the panoptic domain more sensitive to pixel level errors than semantic segmentation, such as around object boundaries. We see this especially when we add time series frames (discussed in Sec. A1.4), with the relative performance dropping as more frames are added. This is expected, as alignment errors will accumulate, espe-

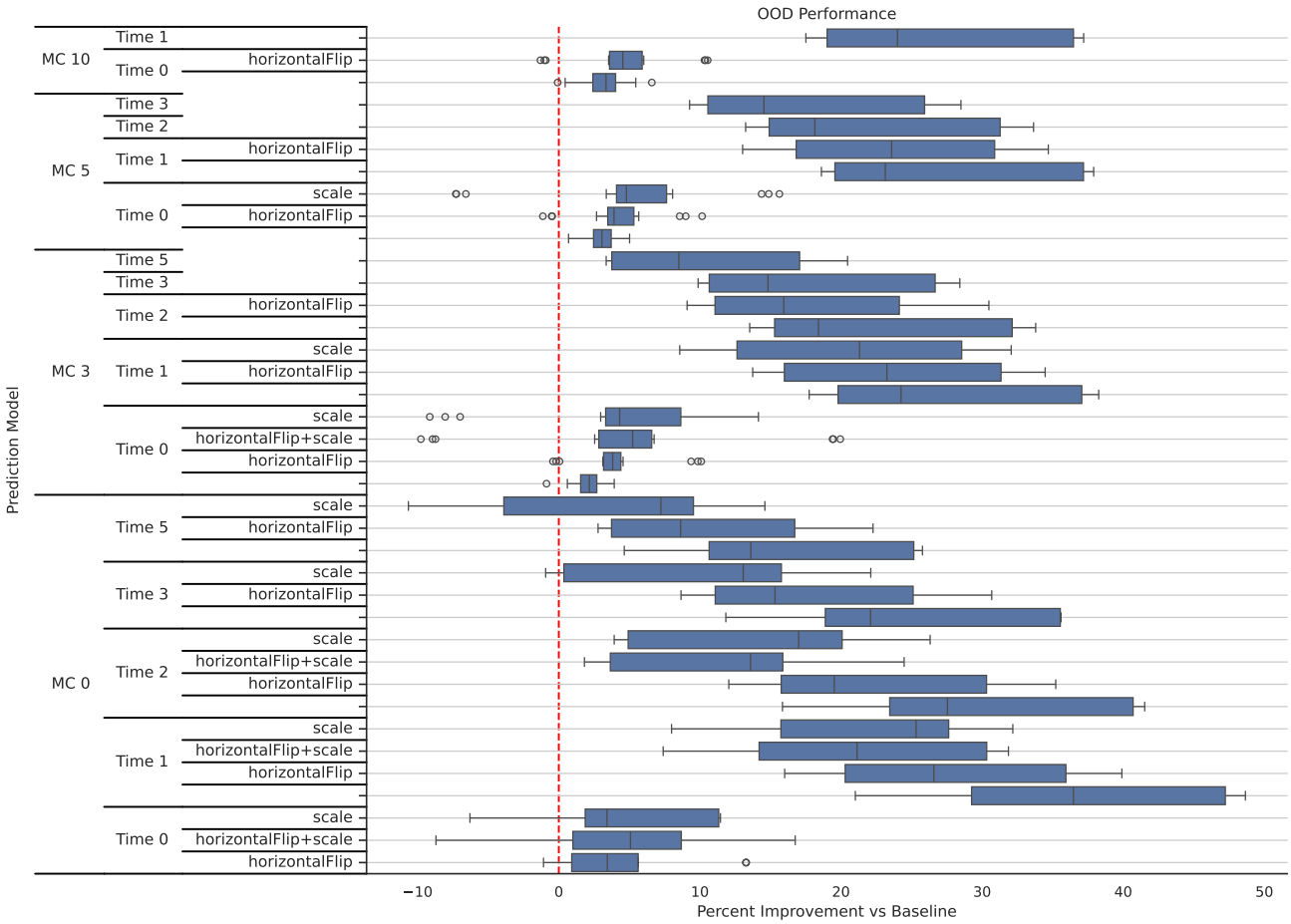


Figure A5. Results on the downstream task (**V8**) of out-of-distribution detection, plotted by prediction model (**V3**) and broken down into each individual tested configuration. Experiment parameters follow Fig. 3a, except a traditional box plot is used.

cially at object boundaries. This is also the case with ground truth optical flow data due to its deficiencies, as discussed in Sec. A1.2.

A8. Computational Cost

Training models is fairly resource intensive, and our requirements follow those of [1] where a multi-GPU machine with sufficiently large GPUs to fit larger batch sizes is ideal. In our work, we primarily used nodes with 4 Nvidia V100-SMX2-32GB.

As our work focuses on uncertainty estimation, inference performance is an issue, particularly with ensembles requiring multiple samples. Our implementation of ensembles is straightforward but restricted to a single GPU. Different experiments, however, such as with a different seed or dataset can of course run concurrently on other GPUs. Depending on the size of the dataset validation set and prediction model, a pass through can take anywhere from a few minutes for a deterministic baseline to several hours for prediction model configurations with many samples. At inference time, with Mask Distance sample aggregation all experiments in the configurations presented can run on a single 12GB GPU. To evaluate all the approaches we show in this study, however, multiple GPUs are required to run experiments in parallel so as to have them finish in a reasonable amount of time. As implemented, we also save the network output to disk (segmentation results, uncertainty estimates) for later processing which does not require a GPU (*e.g.* calculating the AURC metrics for failure detection) so as to allow the GPUs to continue with other experiments but this requires multiple terabytes of storage space, ideally over 100 TB.

One of the key limitations of any ensemble approach is the time required to compute samples, and our approach is no different. To demonstrate, we evaluate different prediction model combinations on the Cityscapes dataset with the ResNet50 backone and record the inference time required per image to generate the samples, aggregate them and calculate uncertainty measures. Time to load data from disk is excluded. Evaluation is done on a single machine with

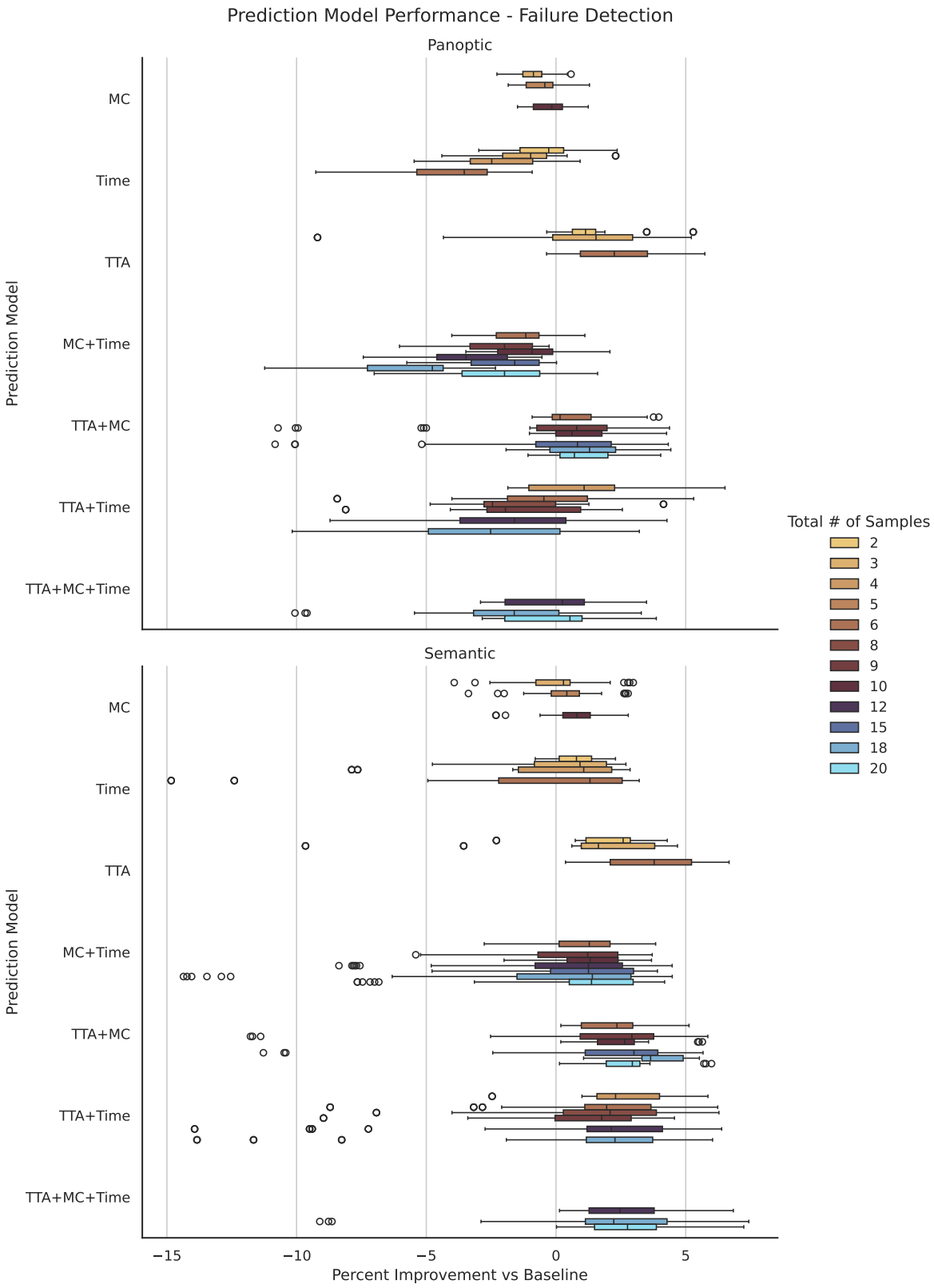


Figure A6. Results on the downstream tasks (**V8**) of failure detection, following the parameters of Fig. 2a except we break out the configurations by the number of samples and use a traditional box plot.

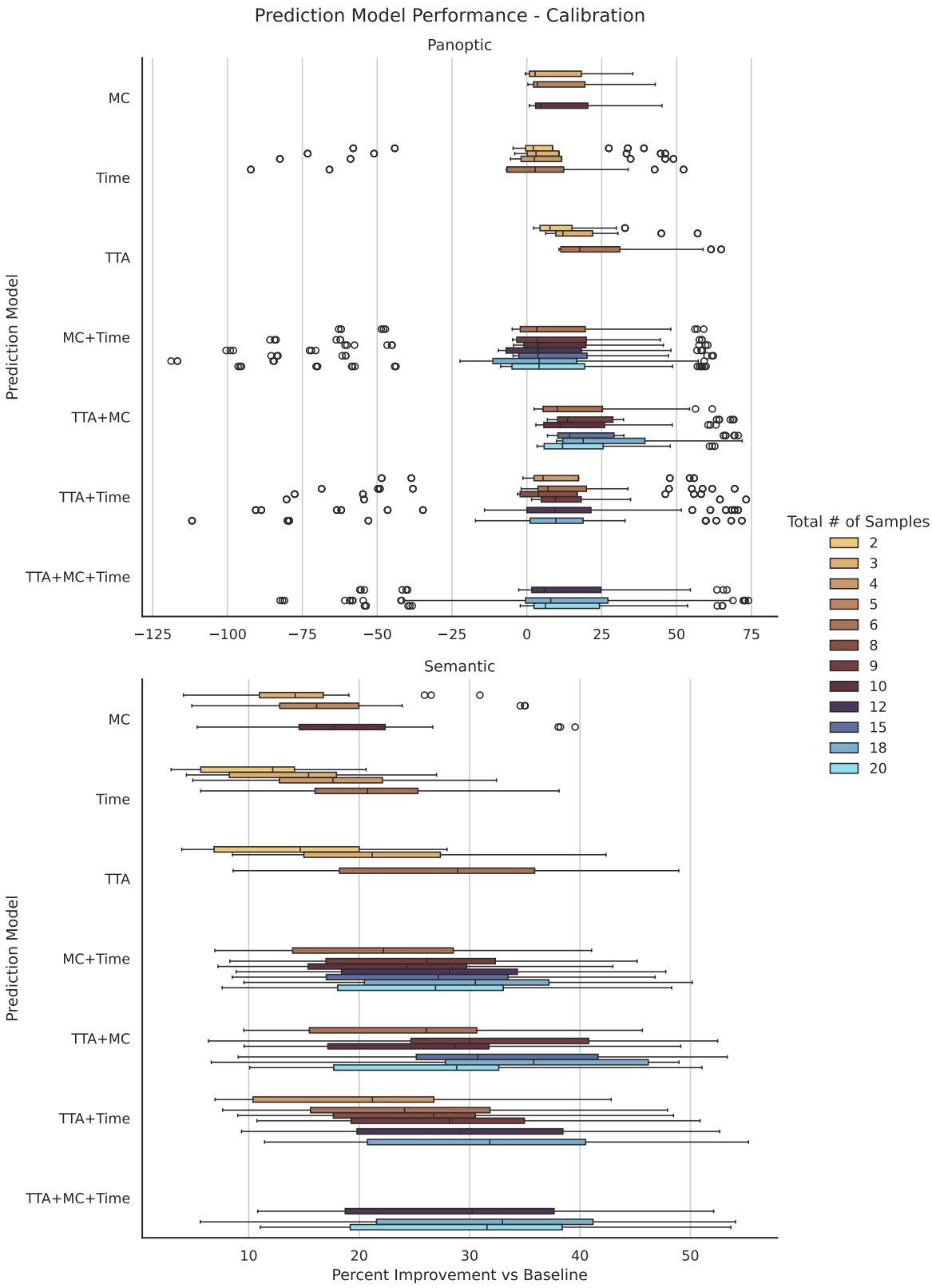


Figure A7. Results on the downstream tasks (**V8**) of calibration, following the parameters of Fig. 2a except we break out the configurations by the number of samples and use a traditional box plot.

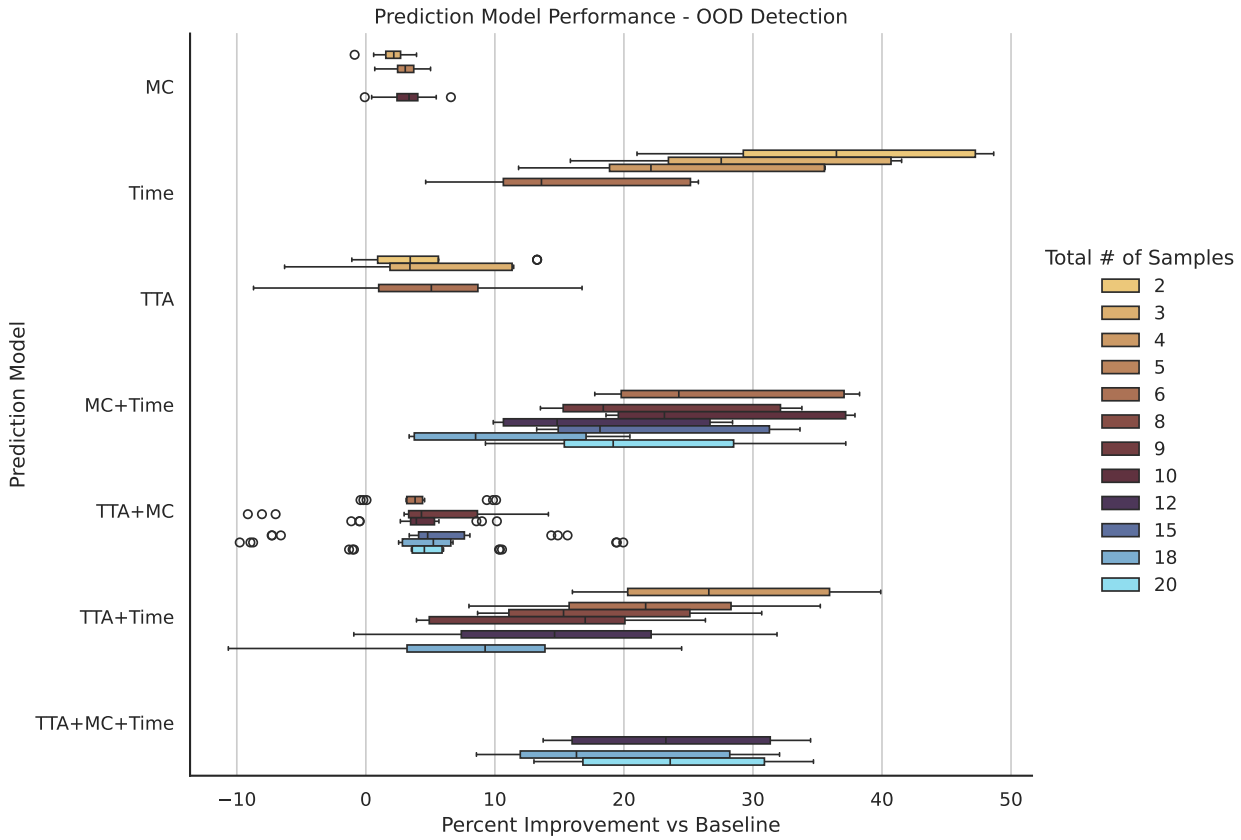


Figure A8. Results on the downstream tasks (**V8**) of out-of-distribution detection, following the parameters of Fig. 3a except we break out the configurations by the number of samples and use a traditional box plot.

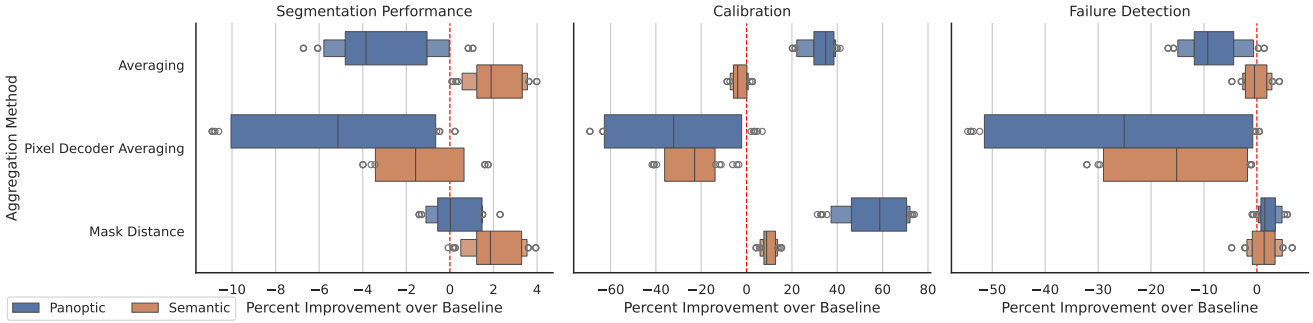
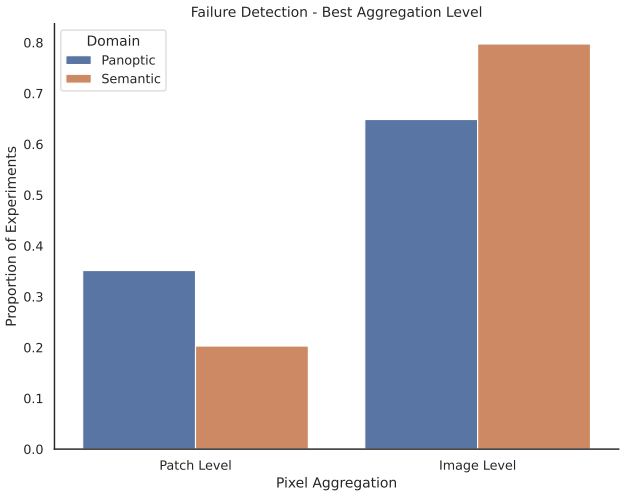


Figure A9. Comparison of different approaches for **Sample Aggregation** (**V4**) in terms of performance on segmentation and the downstream tasks (**V8**) of calibration and failure detection. As in Fig. 2, letter-value plots are used. Dataset and backbone (**V1** and **V2**) are fixed to *Cityscapes* and *ResNet50* respectively. The baseline is a deterministic model with no sampling and thus sample aggregation (**V4**) is not needed. Prediction model combinations (**V3**) are swept through, while **V6** and **V7** are optimized over.

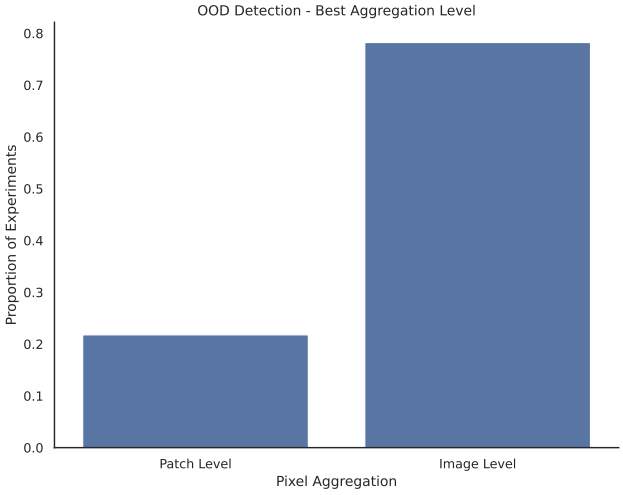
a Nvidia RTX 4090 with 24B of VRAM. All prediction model combinations applicable in Tab. A4 are tested, with the exception of the following with the Averaging sample aggregation approach due to VRAM limits:

- 3 MC samples, 1 previous time frame, horizontal flips for TTA
- 3 MC samples, 1 previous frame, scale transforms for TTA
- No MC samples, 1 previous frame, horizontal flips and scale transforms for TTA
- No MC samples, 2 previous frames, horizontal flips and scale transforms for TTA

In Figs. A12 to A14 we compare the inference time required for the Mask Distance, Pixel Decoder and Averaging approaches respectively. We see that the number of samples



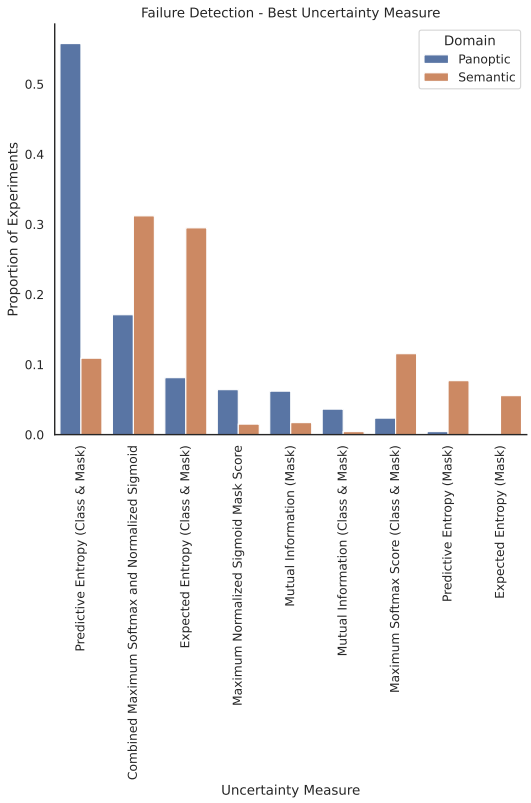
(a)



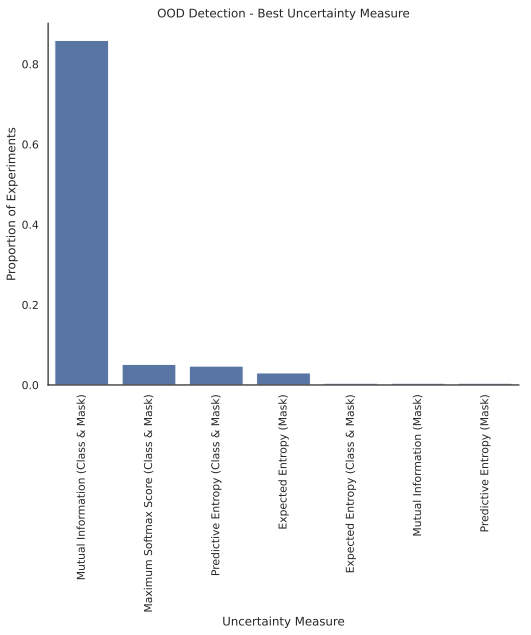
(b)

Figure A10. Pixel Aggregation (**V7**) comparison, presented as a proportion of experiments where one approach was better than the other. Experiments used are the same as Figs. 2 and 3 for (a) failure detection and (b) out-of-distribution detection respectively. Comparison is made after fixing dataset, backbone, prediction model configuration and uncertainty measure. The seed variable is removed via averaging.

is the most significant factor, with the time required being roughly proportional to the number of samples. We also see that the Averaging approach from [18] is the slowest, followed by our Mask Distance approach. The fastest approach is the Pixel Decoder which we introduce in Sec. A3.2, with the required time for similar prediction model configurations being a fraction of the two other approaches. As with all other variables we discuss, a practitioner deploying a model in the real world will need to make a choice as to which sample aggregation may suit them best in the trade-off between uncertainty estimation quality and speed.



(a)



(b)

Figure A11. Uncertainty measure (**V6**) comparison, presented as a proportion of experiments where one approach was better than the other. Experiments used are the same as Figs. 2 and 3 for (a) failure detection and (b) out-of-distribution detection respectively. Comparison is made after fixing dataset, backbone, and prediction model configuration. The seed variable is removed via averaging.

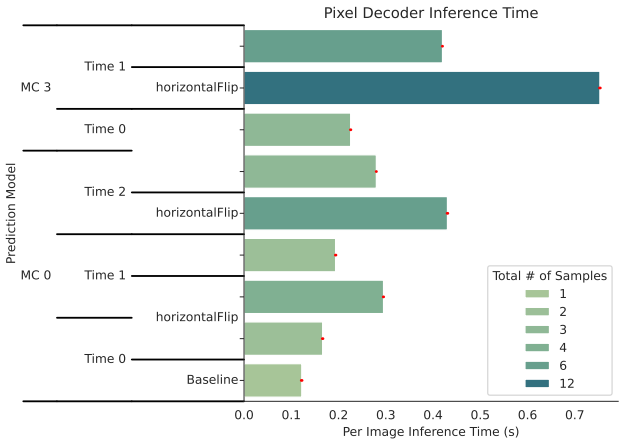


Figure A12. Inference time comparison for the Pixel Decoder approach to sample aggregation (**V4**), executed on a Nvidia RTX 4090. Results are averaged over 3 runs; error bars in red show the 95% confidence interval, generated via bootstrapping.

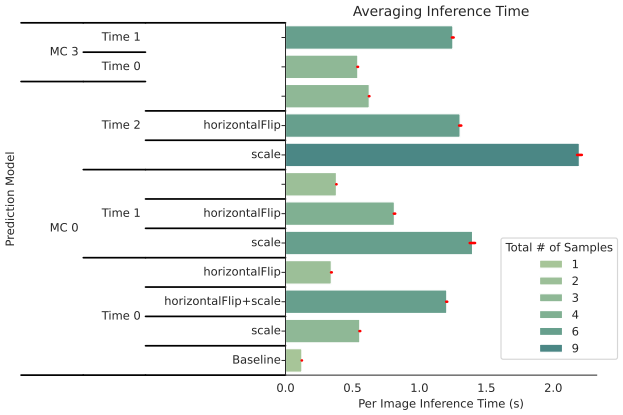


Figure A13. Inference time comparison for the Averaging approach [18] to sample aggregation (**V4**), executed on a Nvidia RTX 4090. Results are averaged over 3 runs; error bars in red show the 95% confidence interval, generated via bootstrapping.

Supplementary Material References

[1] Bowen Cheng, Ishan Misra, Alexander G. Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention Mask Transformer for Universal Image Segmentation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1280–1289, 2022. A-2, A-3, A-4, A-5, A-10

[2] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. A-1

[3] Jacob Deery, Chang Won Lee, and Steven L. Waslander. ProPanDL: A Modular Architecture for Uncertainty-Aware Panoptic Segmentation. In *2023 20th Conference on Robots and Vision (CRV)*, pages 137–144, 2023. A-5, A-7

[4] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D. Cubuk, Quoc V. Le, and Barret Zoph. Simple Copy-Paste Is a Strong Data Augmentation Method for Instance Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2918–2928, 2021. A-5

[5] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1321–1330. PMLR, 2017. A-3

[6] Po-Yu Huang, Wan-Ting Hsu, Chun-Yueh Chiu, Ting-Fan Wu, and Min Sun. Efficient Uncertainty Estimation for Semantic Segmentation in Videos. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. A-2

[7] Paul F. Jaeger, Carsten Tim Lüth, Lukas Klein, and Till J. Bungert. A Call to Reflect on Evaluation Practices for Failure Detection in Image Classification. In *The Eleventh International Conference on Learning Representations*, 2023. A-6

[8] Kim-Celine Kahl, Carsten T. Lüth, Maximilian Zenk, Klaus Maier-Hein, and Paul F. Jaeger. ValUES: A Framework for Systematic Validation of Uncertainty Estimation in Semantic Segmentation. In *The Twelfth International Conference on Learning Representations*, 2024. A-4, A-5, A-6, A-9

[9] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollar. Panoptic Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. A-6

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing. A-3

[11] Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Ann Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the Calibration of Modern Neural Networks. In *Advances in Neural Information Processing Systems*, 2021. A-3

[12] Doug Morrison, Anton Milan, and Nontas Antonakos. Estimating uncertainty in instance segmentation using dropout sampling. In *CVPR 2019 Robotic Vision Probabilistic Object Detection Challenge*, 2019. A-5

[13] Jishnu Mukhoti, Andreas Kirsch, Joost van Amersfoort, Philip HS Torr, and Yarin Gal. Deep deterministic uncertainty: A simple baseline. *arXiv preprint arXiv:2102.11582*, 2021. A-6

[14] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research*, 2024. A-2, A-3

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss,

V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011. A-6

[16] Stephan R. Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for Benchmarks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2232–2241, 2017. A-1

[17] Kshitij Sirohi, Sajad Marvi, Daniel Büscher, and Wolfram Burgard. Uncertainty-Aware Panoptic Segmentation. *IEEE Robotics and Automation Letters*, 8(5):2629–2636, 2023. A-5, A-6

[18] Michael Smith and Frank Ferrie. Uncertainty estimation in deep learning for panoptic segmentation. *arXiv preprint arXiv:2304.02098*, 2023. A-3, A-4, A-6, A-14, A-15

[19] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In *Computer Vision – ECCV 2020*, pages 402–419, Cham, 2020. Springer International Publishing. A-1

[20] Tuan-Hung Vu, Eduardo Valle, Andrei Bursuc, Tommie Kerssies, Daan de Geus, Gijs Dubbelman, Long Qian, Bingke Zhu, Yingying Chen, Ming Tang, Jinqiao Wang, Tomáš Vojíř, Jan Šochman, Jiří Matas, Michael Smith, Frank Ferrie, Shamik Basu, Christos Sakaridis, and Luc Van Gool. The BRAVO Semantic Segmentation Challenge Results in UNCV2024. In *Computer Vision – ECCV 2024 Workshops*, pages 290–306, Cham, 2025. Springer Nature Switzerland. A-3

[21] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: Deformable Transformers for End-to-End Object Detection. In *International Conference on Learning Representations*, 2021. A-4

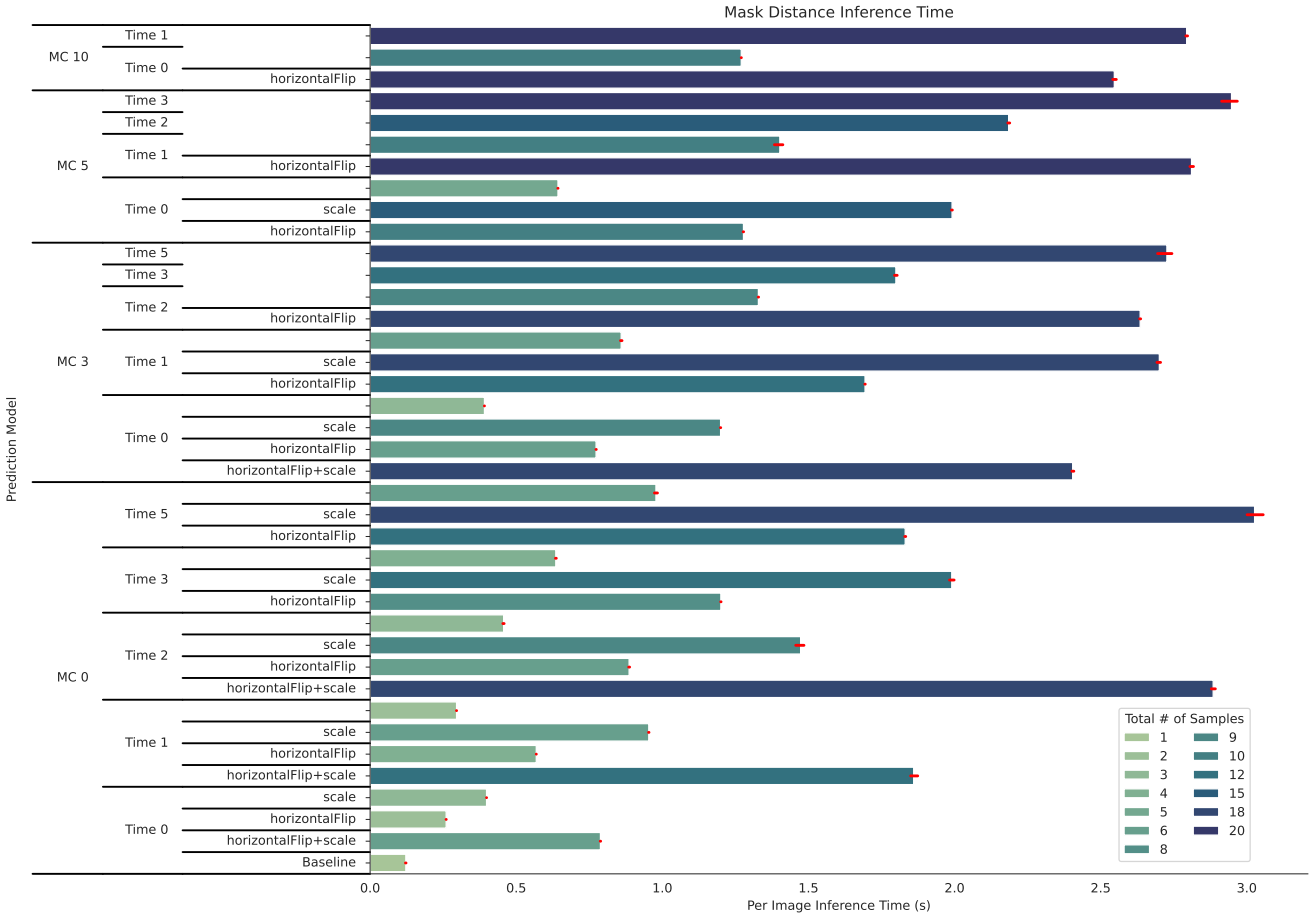


Figure A14. Inference time comparison for the Mask Distance approach to sample aggregation (V4), executed on a Nvidia RTX 4090. Results are averaged over 3 runs; error bars in red show the 95% confidence interval, generated via bootstrapping.