On the Viability of Monocular Depth Pre-training for Semantic Segmentation: Supplementary Material

Dong Lao[1], Fengyu Yang[2], Daniel Wang[2], Hyoungseob Park[2], Samuel Lu[1], Alex Wong[2], and Stefano Soatto[1]

[1] UCLA Vision Lab, Los Angeles, CA 90024, USA
{lao, soatto}@cs.ucla.edu, samuellu@ucla.edu
[2] Yale Vision Lab, New Haven, CT 06511, USA
{fengyu.yang, daniel.wang.dhw33, hyoungseob.park, alex.wong}@yale.edu

A Formalization of Empirical Set-up

The Information Bottleneck Lagrangian in Sect. 3 cannot be computed because we do not have access to the analytical form of the joint distributions of the variables $(\mathbf{x}, \mathbf{h})$, $(\mathbf{h}, \mathbf{y})$ and $(\mathbf{x}, \hat{\mathbf{h}})$, $(\hat{\mathbf{h}}, \mathbf{y})$. Even defining, let alone computing, the Shannon Information for random variables that are deterministic maps $\phi(\cdot)$ of variables $\mathbf{h}$ defined in the continuum is non-trivial [1]. It is possible, however, to bound the Information Bottleneck, which is not computable, with the *Information Lagrangian* [2], which only depends on the datasets $\mathcal{D}_y, \mathcal{D}_z$ and $\mathcal{D}_s$. However, the bound depends on constants that are functions of the complexity of the datasets, which are different for different tasks, which would render them useless in answering the question in (3). Therefore, we consider the validation error as a proxy of residual information:

$$L_z(w''|w) = \sum_{\mathbf{h}^n = \phi_w(\mathbf{x}^n)} -\log p_{w''}(\mathbf{y}^n|\mathbf{h}^n) \simeq H(\mathbf{y}|\mathbf{h}) \tag{1}$$

for pre-training using depth estimation, and

$$L_y(w''|w') = \sum_{\hat{\mathbf{h}}^n = \phi_{w'}(\mathbf{x}^n)} -\log p_{w''}(\mathbf{y}^n|\hat{\mathbf{h}}^n) \simeq H(\mathbf{y}|\hat{\mathbf{h}}) \tag{2}$$

for pre-training using another pre-training method. The complexity terms $I(\mathbf{h}; \mathbf{x})$ and $I(\hat{\mathbf{h}}; \mathbf{x})$ are minimized implicitly by the capacity control mechanisms in the architecture (*i.e.,* the maps $\phi_w(\cdot|\mathcal{D}_z)$ and $\phi_{w'}(\cdot|\mathcal{D}_y)$), for instance pooling; in the regularizers, for instance weight decay and data augmentation [1]; and in the optimization, for instance stochastic gradient descent [3]. The losses above are computed by summing over the samples in the validation set $\mathcal{D}_s = \{\mathbf{x}^n, \mathbf{y}^n\}$.

B Implementation Details

B.1 Warping

$$\hat{x}(i,j) = x \circ \pi_{g,z}^{-1}(i,j) \tag{3}$$

is the *warping* of an image x onto the image plane of another camera related to it by a change of pose $g \in SE(3)$, through the depth map z, via a *reprojection map* π^{-1}

$$\pi^{-1}(i,j) = K_+ \pi(R_t K^{-1}[i,j,1]^T z(i,j) + T_t) \tag{4}$$

that embeds a pixel (i,j) in homogeneous coordinates, places it in the camera reference frame via a calibration matrix K, back-projects it onto the scene by multiplying it by the depth $z(i,j) = \phi_w(x(i,j)|\mathcal{D}_z)$ and then transforming it to the reference frame of another camera with a rigid motion $g = (R,T)$, where the rotation matrix $R \in SO(3)$ and the translation vector $T \in \mathbb{R}^3$ transform the coordinates of spatial points $P \in \mathbb{R}^3$ via $P \mapsto RP + T$. Here π is a canonical perspective projection $\pi(P) = [P(1),\ P(2)]/P(3)$ and the calibration map K_+ incorporates quantization into the lattice. Here, we assume that the intrinsic calibration matrix K is known, otherwise it can be included among nuisance variables in the optimization along with the inter-frame pose g_t when minimizing the reprojection error ℓ in (3).

Note that the reprojection error [5, 19–21] could be minimized with respect to w, which is shared among all images and yields a depth map through $z_t = \phi_w(x_t|\mathcal{D}_z)$, or directly with respect to z_t in (4), which does not require any induction. Since the goal of pre-training is to capture the inductive bias we adopt the former and discuss it in detail in Sect. 5.

B.2 Training and Evaluation Details on KITTI

Pre-training for unsupervised depth estimation. Monodepth2 is trained by optimizing a linear combination of photometric reprojection error and an edge-aware local smoothness prior

$$L(w') = w_{ph}\ell_{ph} + w_{sm}\ell_{sm}, \tag{5}$$

where

$$\ell_{ph} = \sum_{i,j,n,t} (1 - \text{SSIM}(x^n_{t+1}(i,j), \hat{x}^n_t(i,j))) + \alpha|x^n_{t+1}(i,j) - \hat{x}^n_t(i,j)|_1 \tag{6}$$

$\hat{x}_t$ is the warped image (3) and ℓ_{sm} is the edge-aware smoothness prior

$$\ell_{sm} = \sum_{i,j,n,t} |\partial_X z^n_t(i,j)| e^{-|\partial_X x^n_t(i,j)|} + |\partial_Y z^n_t(i,j)| e^{-|\partial_Y x^n_t(i,j)|}, \tag{7}$$

w_{ph} and w_{sm} are hyper-parameter weights.

Semantic segmentation fine-tuning. Following the notation in Sect. 3 in the main paper, we denote with $\phi_{w''} : \mathbf{x} \mapsto \hat{\mathbf{y}}$ the semantic segmentation network

to be fine-tuned, which maps an image $\mathbf{x}$ to a semantic label $\mathbf{y}$. Note that $\phi_{w'}$ (classification network) is parameterized by weights w' of the 'encoder' network, and ϕ_w (depth network) is parameterized by weights w of both the 'encoder' and a 'decoder'. When pre-trained for classification, we initialize the encoder part of w'' by w' and the decoder part by random weights; when pre-trained for depth, we initialize both encoder and decoder in w'' using w, except for the last layer, where we change to a randomly initialized fully-connected layer with soft-max. During semantic segmentation fine-tuning, we update w'' (see equation (11) and (12) in the main paper) by minimizing the cross entropy loss

$$L(w''|\bullet) = \sum_{i,j,n,k} -\log(\hat{y}^n(i,j)))\mathbb{1}(y^n(i,j)=k) \tag{8}$$

where i, j are the pixel coordinates, n is the number of images in the training set, k is the class label, $\hat{y} = \phi_{w''}(\mathbf{x}^n)$ is the network output. This is implemented via the Negative Log Likelihood (NLL) loss in Pytorch. Under different experimental settings, we either update all parameters in w'' or only the decoder part of w'' to minimize (8).

KITTI contains 21 semantic classes, and we fine-tune on all of them. However, since we do not use a separate dataset with segmentation labels, many of the semantic classes are seldom seen (*e.g.*, 'train', 'motorcycle'), especially with just 16 training images. In such cases, these classes always receive zero IoU, which downweights the mIOU metric (but still yields high P.Acc). Therefore, we compute mIoU on a subset of 7 representative classes unless stated otherwise. The results of 21 classes exhibit the same trends.

Image normalization. In fine-tuning, we apply the same image normalization that is consistent with the pre-training step. If the network is pre-trained by ImageNet classification, we normalize the image values by mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]; if pre-trained by Monodepth2, we normalize the image values to [0,1]; we also normalize to [0,1] when training from random initialization.

Optimizer. After a grid search, we choose 1×10^{-5} as the initial learning rate for the ADAM optimizer. The learning rate is updated by a standard cosine learning rate decay schedule in every iteration.

B.3 Details for training optical flow

To train a neural network f_θ parameterized by θ to estimate optical flow for a pair of images (x_t, x_{t+1}) from time step t to $t+1$, we leverage the photometric reconstruction loss [12, 14] by minimizing a color consistency term and a structural consistency term (SSIM) between an image $x_{t+1}$and its reconstruction $\hat{x}_t$ given by the warping x_t with the estimated flow $f_\theta(x_t, x_{t+1})$:

$$\ell_{ph} = \sum_{n,i,j,t} \lambda_{co}\big(|x^n_{t+1}(i,j) - \hat{x}^n_t(i,j)|\big) + \lambda_{st}\big(1 - SSIM(x^n_{t+1}(i,j), \hat{x}^n_t(i,j))\big), \tag{9}$$

where $\hat{x}_t = x_t \circ f_\theta(x_t, x_{t+1})$, $f_\theta(\cdot) \in \mathbb{R}^{2HW}$ and $\lambda_{co} = 0.15$, $\lambda_{st} = 0.15$ are the weights for color consistency and SSIM terms for L_{ph}, respectively.

Additionally, we minimize an edge-aware local smoothness regularizer:

$$\ell_{sm} = \lambda_{sm} \sum_{n,i,j} \lambda_X(i,j)|\partial_X f_\theta(x^n_t, x^n_{t+1})(i,j)| + \lambda_Y(i,j)|\partial_Y f_\theta(x^n_t, x^n_{t+1})(i,j)| \tag{10}$$

where $\lambda_{sm} = 5$ is the weight of the smoothness loss, ∂_X, ∂_Y are gradients along the x and y directions, and the loss for each direction is weighted by $\lambda_X := e^{-|\partial_X x^n_t|}$ and $\lambda_Y := e^{-|\partial_Y x^n_t|}$ respectively.

We trained f_θ using Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We set the initial learning rate to be 5×10^{-4} for the first 25 epochs and decreased it to 5×10^{-5} another 25 epochs. We used a batch size of 8 and resized each image to 640×192. Because KITTI has two video streams from left and right stereo camera, we randomly sample batches from each stream with a 50% probability. Training takes ≈ 10 hours for ResNet18 backbone and ≈ 20 hours for ResNet50.

B.4 Details for training on Cityscapes

Different from KITTI, we perform pre-training for depth on a modified DeepLabV3 architecture. We change the prediction head to output a single depth channel instead of 19 classification channels. We follow the standard image normalization of ImageNet training. We use ADAM optimizer with an initial learning rate of 1e-4 and linear learning rate decay. Data augmentations include brightness, contrast, saturation and random horizontal flips. Each model is trained on a single Nvidia GeForce GTX 1080 Ti GPU with a batch size of 8 for 15000 iterations, which takes approximately 6.5 hours. During fine-tuning, we re-initialize the prediction head for segmentation. Each model is trained on two Nvidia GeForce GTX 1080 Ti GPUs. We use SGD optimizer with polynomial learning rate decay. We did a learning rate search and report the best performing settings. We find using 0.1 as the initial learning rate yields optimal results when pre-trained by depth, while 0.01 works the best with ImageNet pre-training. All other settings follow the original DeepLabV3 implementation. Limited by the GPU memory we crop training images to 512×512 patches and set the batch size to be 8. Each model is trained for 60000 iterations (30000 iterations under 'controlled' settings), taking $\tilde{1}5$ hours, and we report the accuracy after the final epoch.

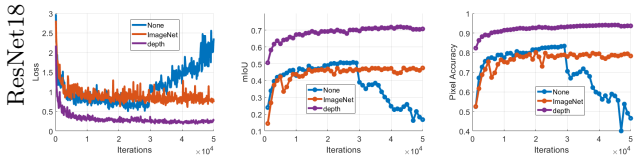

Fig. 1: Training ResNet with high resolution images. *Similar to low resolution images, pre-training on depth also improves semantic segmentation accuracy on high resolution images.*

B.5 Details for training on NYU-V2

Because NYU-V2 provide image and depth map pairs, similar to Cityscapes, we directly train Monodepth2 ϕ_w by minimizing an L_1 loss:

$$\ell_{L_1} = \sum_{n,i,j} \mathbb{1}(z^n(i,j) > 0)\big(|\phi_w(x^n)(i,j) - z^n(i,j)|\big), \tag{11}$$

where $\phi_w(x)$ is the predicted depth for an image x and z is the ground truth depth from a Microsoft Kinect. Because the ground truth is only semi-dense, this loss is only computed where there is valid depth measurements *i.e.*, $z(i,j) > 0$.

We trained Monodepth2 using the ADAM optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We set the initial learning rate to be 1×10^{-4} for 5 epochs and decreased it to 5×10^{-5} another 5 epochs for a total of 10 epochs. We used a batch size of 8 and resized each image to 448×384. For data augmentation, we perform random brightness, contrast and saturation adjustments within the range of $[0.80, 1.20]$ with a 50% probability. Pre-training the depth network takes $\approx$ 19 hours for ResNet18 backbone and $\approx$ 30 hours for ResNet50. After pre-training, we do semantic segmentation fine-tuning on the training set. As with KITTI, we apply the ADAM optimizer with 1×10^{-5} initial learning rate and cosine learning rate decay, and restrict data augmentation to horizontal flipping. Limited by GPU memory, for ResNet18 we use batch size 32, and for ResNet50 we use batch size 16. All models are trained for 20000 iterations. Each experiment is repeated by four independent trials.

C Experimental Results (Continued)

C.1 Result on high-resolution images (KITTI)

In Fig. 1 we show results trained with full resolution (1024×320) and full dataset (200 images), in which case we train for an extended number of iterations (50000). Pre-training by depth still outperforms ImageNet pre-training and random initialization. Note that in this case using random initialization is unstable and training diverges after 30000 iterations.

We also conduct experiments on high-resolution images. Results are consistent with low-resolution experiments. Although both ImageNet and depth initialization converged to the same level of accuracy, the depth pre-trained model

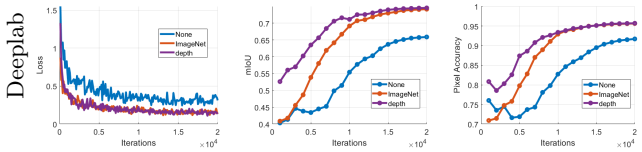

Fig. 2: Training Deeplab on high resolution images. *Compared to lower resolutions, performance improves for different initializations. Training loss (left) is similar between ImageNet and depth initialization, but mIOU (center) and pixel accuracy (right) are higher for depth initialization for similar loss values.*

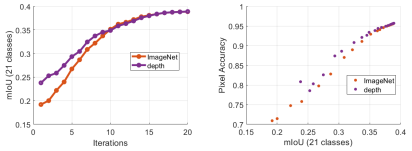

Fig. 3: mIoU v.s. pixel accuracy during training DeepLab. *At each same level of mIoU, model pre-trained by depth has a higher pixel accuracy. This validates our conjecture that the depth-pre-trained model learns 'bigger' classes faster.*

shows a high accuracy in earlier iterations. This is interesting given that training loss of both ImageNet and depth pre-train are almost the same. We conjecture that the features learned by single-image depth estimation are more conducive to segmenting 'bigger' classes (e.g. 'road', 'building') which are mostly rigid and take up larger portions of the image.

To further investigate this behavior, in Figure 3 we plot the mIoU curve on all (21 classes), and a scatter plot of mIoU versus pixel accuracy in the training process. While mIoU on all 21 classes follows the trend observed on 7 classes, the scatter plot shows that at each same level of mIoU, models pre-trained by depth have higher pixel accuracy. This validates our conjecture that the depth-pre-trained model learns 'bigger' classes faster, since higher performance on these classes will result in high pixel accuracy as they have more pixels.

C.2 Training loss on Cityscapes

In Fig. 4 we show training loss of early iterations on Cityscapes shows that training on small crops of the image improves convergence. We revisit the question we posed in Table 4 in the main text regarding this phenomenon below.

C.3 Visualizations of Segmentation

Fig. 5 shows head-to-head comparisons of representative outputs from semantic segmentation models that have been pre-trained with ImageNet classification, and pre-trained with monocular depth estimation. As discussed in Sect. 4.1 Neural Activations, pre-training on ImageNet biases the model towards capturing

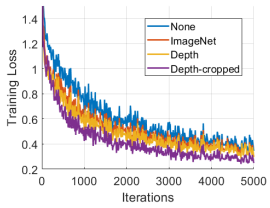

Fig. 4: Training loss of early iterations on Cityscapes. *In early training iterations, the loss of Depth-cropped decreases significantly faster than other initializations.*

generic textures exhibited in the image rather than that object shape (see Fig. 6). This results in a loss of details when fine-tuning for the downstream segmentation task where the goal is precisely to capture object boundaries. We illustrate the drawback of pre-training on ImageNet in Fig. 5-left where the model over-predicts the street sign in the center (highlighted in yellow) and under-predicts the pedestrians on the right (highlighted in red) with spurious predictions of the vehicle class alongside them. This is in contrast to pre-training with monocular depth estimation, where the downstream segmentation model is able to capture the edge between the street sign and building regions in the middle as well as the small pedestrian regions in the far distance on the right. Additionally, we show in Fig. 5-right that an ImageNet pre-trained model has difficulty outputting the same class for a consistent surface like the sidewalk on the left, which is unlike a model pre-trained on depth. This is also supported by our results in Appendix C and Sect. 4.1 in the main text where the "larger" object classes tend to be more easily learned (higher general P.Acc) by a model pre-trained on depth than one trained on ImageNet.

While classification is a semantic task, training for it requires discarding nuisances including objects (other than one that is front and center, see neural activations in Fig. 6 and results with a frozen pre-trained encoder in Fig. 4) that may exist in the background of an image. Pre-training for depth, on the other hand, involves solving correspondence problems, which naturally requires estimating the boundaries of objects and consistent locally-connected, piecewise smooth surfaces. Thus, one may hypothesize that pre-training to predict depth or geometry process makes it more straightforward to assign these surfaces with a semantic class or label i.e. road, pedestrian, building. We conjecture that this may play a role in sample complexity as illustrated by the performance improvements observed over ImageNet pre-training when training with fewer samples (Fig. 3).

C.4 Additional Results on Object Detection

To demonstrate the versatility of depth pretraining, we extend our experiments to object detection. Using DepthAnything as the depth pre-training method, we conducted a comparison with DINO-V2. The results are shown in Tab. 1. Our

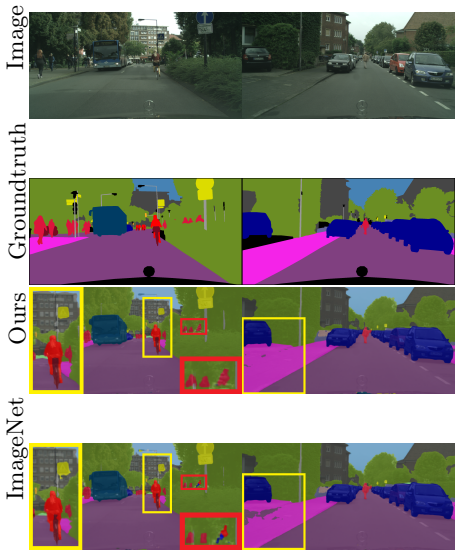

Fig. 5: Head-to-head comparison between ImageNet and monocular depth estimation pre-training for semantic segmentation on Cityscapes. *We visualize the representative outputs of segmentation models pre-trained on monocular depth (row 3) and ImageNet (row 4). Pre-training with depth enables sharper object boundaries, i.e. street sign highlighted in yellow, pedestrians highlighted in red, and more consistent class predictions on large objects like the sidewalk on the left in the right column. We note that pre-training on ImageNet also yields spurious predictions like the vehicle class next to the pedestrians (highlighted in red) and the street sign class on the street light in the middle of the left image. Better viewed zoomed-in and in color.*

findings reveal that depth as pre-training outperforms DINO-V2 on the ADE20K and Cityscapes datasets, while yielding comparable performance on COCO. This outcome is anticipated because DINO-V2's pre-training dataset is object-centric, aligning closely with COCO's characteristics. In contrast, ADE20K and Cityscapes datasets exhibit a bigger domain gap, suggesting that depth pre-training may generalize to diverse data domains. We note that this experiment only marks the tip of the iceberg; related tasks include moving object detection [11, 13], which we leave for future work.

D Extended Discussion

D.1 Cropping Size During Pre-training

One may object that once a depth map has been estimated, a semantic map is just a matter of aligning labels. However, depth is not necessarily a piecewise constant function, although it is generally piecewise smooth. So, for instance, the road at the bottom center of KITTI images is a slanted plan that does not

Table 1: Quantitative results of object detection. *We initialize a Vit-L for MaskRCNN using DINOv2 and DepthAnything pretrained weights on three object detection datasets. Depth pretraining improves on ADE20K and Cityscapes while being comparable on COCO.*

	ADE20K			Cityscapes			COCO		
Pretraining	mAP↑	mAP@50↑	mAP@75↑	mAP↑	mAP@50↑	mAP@75↑	mAP↑	mAP@50↑	mAP@75↑
DinoV2	0.326	0.509	0.354	0.321	0.540	0.310	**0.499**	0.719	**0.540**
Depth	**0.334**	**0.521**	**0.355**	**0.327**	**0.548**	**0.332**	**0.499**	**0.720**	0.539

correspond well to any constant value, yet the model converts it into a consistent class.

One may also object that slanted planes at the bottom center of the visual field have a strong bias towards being labeled "road" given the data on which the model is trained.

For this reason, we conducted the experiments shown in Tab. 5, whereby we select random crops of 3% of the size of the image, and use those for pre-training rather than the full image. This way, there is no knowledge of the location of the patch of the slanted plane relative to the image frame. We were expecting a degradation in performance, but instead observing an improvement.

While in theory full consideration of the visual field is more informative, provided sufficient training data, due to the limited volume of the training set and the strong biases in the training data, breaking the image into smaller patches and discarding their relation (position on the image plane) may help break the spurious dataset-dependent correlations and lead to better generalization after fine-tuning.

This conclusion is speculative, and we leave full testing to future work. The important aspect of this experiment is to verify that fine-tuning semantic segmentation after depth pre-training is not just a matter of renaming a piecewise smooth depth field into a piecewise constant labeled field.

D.2 Relation to Prior Arts

The primary objective of this study is to conduct an exhaustive examination of the adoption of monocular depth as a pre-training technique for semantic segmentation and to compare it with classification (which has been the de-facto approach for weight initialization). It is acknowledged that prior research [6–8] has also recommended the use of depth as a pre-training method. While we are not claiming originality in using depth, our study stands out as the first to systematically investigate this issue under different setups including network architecture, resolution, and supervision. Nevertheless, there are also distinctions between our study and these methods.

[8] performs pre-training by estimating relative depth on images, while we focus on estimating absolute depth either through supervised or unsupervised pre-training. We present results that contradict those of [8]. While [8] claim that

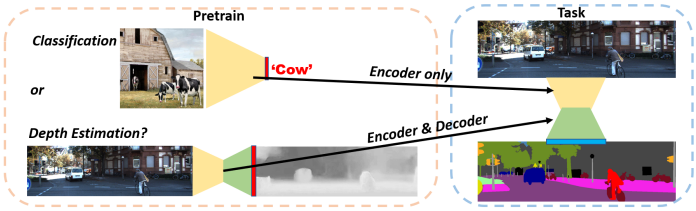

Fig. 6: *Image classification introduces uncontrolled biases when used to pre-train semantic segmentation networks, where one requires an additional decoder. Depth estimation, on the other hand, is not subject to semantic bias in the pre-training dataset, and eliminates the need for human annotation, and can easily adapt the pre-training dataset to the domain of interest. The question is whether such a process can improve performance and reduce dependency of annotated pre-training datasets in fine-tuning for semantic segmentation.*

ImageNet is the better pre-training approach, our findings show better performance when using our depth pretraining as compared to ImageNet initialization alone. The difference in findings is suprising as we observe consistent performance boost across our experiments; one possible reason may be that camera calibration is available to us (also a valid assumption for real-world scenarios that require semantic segmentation, such as self-driving), and it is well-known that un-calibrated depth estimation is a more difficult problem than the calibrated case. Another difference (and the possible reason for the contradicting results) is that [8] synthesizes relative depth from optical flow estimation network, and train a depth prediction network by minimizing L1 difference between predictions and relative depth. The authors also state that potential errors in optical flow will impact (and compound the error in) the downstream depth.

[6, 7] use self-supervised depth prediction as a proxy task for semantic segmentation, where the features are regularized to stay close to ImageNet features. However, our experiments demonstrate that in some cases, ImageNet features may not support the task of semantic segmentation as observed in Fig. 6, and also empirically validated in Fig. 5. Even when we train the depth model with ImageNet initialization, we do not assume that the features should remain largely unchanged. Nonetheless, we are not against using ImageNet, as it is off-the-shelf and can be useful to expedite training depth estimation networks, likely due to the formation of generic filter banks in the early layers.

We would emphasize that our paper validates the findings in prior works [6–8], yet provides a more comprehensive longitudinal study (across different architectures, datasets, supervision, etc.) towards using monocular depth as pre-training for semantic segmentation.

D.3 Insights On the Intuitions and Feasibility of Using Monocular Depth as Pre-training

ImageNet classification is widely regarded as the primary task for pre-training in the context of semantic segmentation. The dataset comprises more than 14

million annotated images, collected through crowd-sourcing with the assistance of around 15,000 paid annotators. Empirical studies have consistently confirmed the advantages of using ImageNet for initial model training, resulting in substantial enhancements in the performance of semantic segmentation tasks. This outcome is unsurprising, as both image classification and semantic segmentation involve understanding the meaning of objects in images. It's important to note that obtaining detailed pixel-level annotations for semantic segmentation is a costly and resource-intensive process. Consequently, datasets specifically tailored for semantic segmentation are considerably smaller in scale compared to ImageNet, often differing by several orders of magnitude in terms of data size.

In contrast, the KITTI dataset, which is widely recognized for its use for depth estimation for driving scenarios, consists of approximately 86,000 training images. These images are captured at a rate of 10 frames per second, resulting in less than three hours of driving video. Collecting this type of driving video data is relatively straightforward and requires only a driver and a dashboard camera. In addition to continuous video data, collecting training data for monocular depth estimation can also utilize hardware, such as multi-view stereo systems or depth sensors like Time-of-Flight (ToF) and Lidar. What is common among these data sources is that they demand minimal labor and resources compared to the extensive efforts needed for ImageNet data collection. This affordability allows for the scaling up of initial model training directly within the domain of interest.

From an intuitive perspective, it is more natural to transfer knowledge between tasks that share semantic similarities, such as between different semantic tasks, than to transfer between tasks with distinct characteristics, like transitioning from a geometric task to a semantic task. The main problem is that image classification is *defined* by induction and therefore does not only entail, but *requires* a strong inductive bias, which opens the door to potentially pernicious side-effects. Induction is required because, continuing the example of the image labeled as "cow", there is nothing in the image of a scene that would enable one to infer the three-letter word "cow." The only reason we can do so is because the present image resembles, in some way implicitly defined by the training process, *different images, of different scenes,* that some human has tagged with the word "cow." However, since images, no matter how many, are infinitely simpler than even a single scene, say the present one, this process is not forced to learn that the extant world even exists. Rather, it learns regularities in images, each of a different scene since current models are trained with data aiming to be as independent as possible, agnostic of the underlying scene.

Now, one may object that monocular depth estimation is itself undecidable. This is why monocular depth estimators are trained with either multiple views (motion or stereo), or with some form of supervision or partial backing from additional modalities, such as sparse depth from lidar [10] or other range sensor [16]. Then, a depth estimate is just a statistic of the learned prior. As a result, depth estimation networks should never produce *one* depth estimate for each image, but rather a distribution over depth maps, conditioned on the given

image [23, 24]. Given an image, every depth map is possible, but they are not equally likely. The posterior over depth given an image then acts as a prior in depth inference using another modality, *e.g.,* unsupervised depth completion. The mode of this conditional prior can then be used as a depth estimate if one so desires, but with the proviso that whatever confidence one may place in that point estimate comes from inductive biases that cannot be validated

One additional objection is that, since depth requires optimization to be inferred, and the choice of the loss function is a form of transductive bias, that is no less arbitrary than inductive bias. But this is fundamentally not the case, for the optimization residual in transductive inference refers to the data *here and now*, and not to data of different images in different scenes. In other words, the optimization residual is informative of the confidence of our estimate, unlike the discriminant from an inductively-trained classifier [4].

In some cases, one enriches (augments) the predictive loss with manually engineered transformation, calling the result "self-supervised learning." Engineered transformations include small planar group transformations like translation, rotation, scaling, reflections, and range transformations such as contrast transformation or colorization. But for such group transformations, learning is not necessary since we know the general form of the maximal invariant with respect to the entire (infinite-dimensional) diffeomorphic closure of image transformations [17]. One exception is occlusion and scale changes, which are not groups once one introduced domain quantization. So, pre-training using masking, ubiquitous in language models, does learn a preudo-invariant to occlusion, but rather than simulating occlusions, one can simply observe them in the data (a video). The only supervision is then one bit, provided temporal continuity: The fact that temporally adjacent images portray the same scene. While one would assume that video prediction as a pretraining task [9, 15, 18, 22] would then yield representations with the same set of invariances to support semantics, surprisingly it does not as evident by our findings in training with videos on optical flow.

In this paper, we aim to bypass the artificial constraints imposed both by supervised classification, and self-supervision. Instead, we simply use videos to pre-train a model for depth estimation, without supervision. Then, one can use supervision to fine-tune the model for semantic segmentation. These two tasks are seemingly antipodal, yet depth estimation outperforms image classification when used as pre-training for semantic segmentation.

This also addresses one last objection that one can move to our thesis, which is that, since ImageNet data is available, it makes sense to use it. What we argue here is that, actually, it does not. The argument is corroborated by evidence: Pre-training on a geometric task improves fine-tuning a semantic task, even when compared with pre-training with a different semantic task. Nonetheless, ImageNet pre-training can be useful to expedite training depth estimation networks, likely due to the formation of generic filter banks in the early layers.

References

1. Achille, A., Paolini, G., Soatto, S.: Where is the information in a deep neural network? arXiv preprint arXiv:1905.12213 (2019)
2. Achille, A., Soatto, S.: Emergence of invariance and disentanglement in deep representations. The Journal of Machine Learning Research **19**(1), 1947–1980 (2018)
3. Chaudhari, P., Soatto, S.: Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. In: 2018 Information Theory and Applications Workshop (ITA). pp. 1–10. IEEE (2018)
4. Der Kiureghian, A., Ditlevsen, O.: Aleatory or epistemic? does it matter? Structural safety **31**(2), 105–112 (2009)
5. Fei, X., Wong, A., Soatto, S.: Geo-supervised visual depth prediction. IEEE Robotics and Automation Letters **4**(2), 1661–1668 (2019)
6. Hoyer, L., Dai, D., Chen, Y., Köring, A., Saha, S., Van Gool, L.: Three ways to improve semantic segmentation with self-supervised depth estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 11130–11140 (2021)
7. Hoyer, L., Dai, D., Wang, Q., Chen, Y., Van Gool, L.: Improving semi-supervised and domain-adaptive semantic segmentation with self-supervised depth estimation. arXiv preprint arXiv:2108.12545 [cs] (2021)
8. Jiang, H., Larsson, G., Shakhnarovich, M.M.G., Learned-Miller, E.: Self-supervised relative depth learning for urban scene understanding. In: Proceedings of the european conference on computer vision (eccv). pp. 19–35 (2018)
9. Jin, B., Hu, Y., Tang, Q., Niu, J., Shi, Z., Han, Y., Li, X.: Exploring spatial-temporal multi-frequency analysis for high-fidelity and temporal-consistency video prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4554–4563 (2020)
10. Kuznietsov, Y., Stuckler, J., Leibe, B.: Semi-supervised deep learning for monocular depth map prediction. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 6647–6655 (2017)
11. Lao, D., Sundaramoorthi, G.: Minimum delay moving object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4250–4259 (2017)
12. Lao, D., Sundaramoorthi, G.: Extending layered models to 3d motion. In: Proceedings of the European conference on computer vision (ECCV). pp. 435–451 (2018)
13. Lao, D., Sundaramoorthi, G.: Minimum delay object detection from video. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 5097–5106 (2019)
14. Lao, D., Wang, C., Wong, A., Soatto, S.: Diffeomorphic template registration for atmospheric turbulence mitigation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 25107–25116 (2024)
15. Lao, D., Zhu, P., Wonka, P., Sundaramoorthi, G.: Flow-guided video inpainting with scene templates. In: Proceedings of the IEEE/CVF international conference on computer vision. pp. 14599–14608 (2021)
16. Smisek, J., Jancosek, M., Pajdla, T.: 3d with kinect. In: Consumer depth cameras for computer vision, pp. 3–25. Springer (2013)
17. Sundaramoorthi, G., Petersen, P., Varadarajan, V., Soatto, S.: On the set of images modulo viewpoint and contrast changes. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 832–839. IEEE (2009)

18. Wang, Y., Wu, J., Long, M., Tenenbaum, J.B.: Probabilistic video prediction from noisy data with a posterior confidence. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10830–10839 (2020)
19. Wong, A., Fei, X., Tsuei, S., Soatto, S.: Unsupervised depth completion from visual inertial odometry. IEEE Robotics and Automation Letters **5**(2), 1899–1906 (2020)
20. Wong, A., Soatto, S.: Bilateral cyclic constraint and adaptive regularization for unsupervised monocular depth prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5644–5653 (2019)
21. Wong, A., Soatto, S.: Unsupervised depth completion with calibrated backprojection layers. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 12747–12756 (2021)
22. Wu, Y., Gao, R., Park, J., Chen, Q.: Future video synthesis with object motion prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5539–5548 (2020)
23. Yang, Y., Soatto, S.: Conditional prior networks for optical flow. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 271–287 (2018)
24. Yang, Y., Wong, A., Soatto, S.: Dense depth posterior (ddp) from single image and sparse range. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3353–3362 (2019)