Cross-Domain Self-supervised Multi-task Feature Learning using Synthetic Imagery

Zhongzheng Ren and Yong Jae Lee
University of California, Davis

Abstract

In human learning, it is common to use multiple sources of information jointly. However, most existing feature learning approaches learn from only a single task. In this paper, we propose a novel multi-task deep network to learn generalizable high-level visual representations. Since multi-task learning requires annotations for multiple properties of the same training instance, we look to synthetic images *to train our network. To overcome the domain difference between real and synthetic data, we employ an unsupervised feature space domain adaptation method based on adversarial learning. Given an input synthetic RGB image, our network simultaneously predicts its surface normal, depth, and instance contour, while also minimizing the feature space domain differences between real and synthetic data. Through extensive experiments, we demonstrate that our network learns more transferable representations compared to single-task baselines. Our learned representation produces state-of-the-art transfer learning results on PASCAL VOC 2007 classification and 2012 detection.*

1. Introduction

In recent years, deep learning has brought tremendous success across various visual recognition tasks [42, 23, 71]. A key reason for this phenomenon is that deep networks trained on ImageNet [12] learn *transferable representations* that are useful for other related tasks. However, building large-scale, annotated datasets like ImageNet [12] is extremely costly both in time and money. Furthermore, while benchmark datasets (e.g., MNIST [38], Caltech-101 [19], Pascal VOC [18], ImageNet [12], MS COCO [40]) enable breakthrough progress, it is only a matter of time before models begin to overfit and the next bigger and more complex dataset needs to be constructed. The field of computer vision is in need of a more scalable solution for learning general-purpose visual representations.

Self-supervised learning is a promising direction, of which there are currently three main types. The first uses visual cues within an image as supervision such as recovering the input from itself [67, 26], color from grayscale [73, 74], equivariance of local patchs [49], or predicting the relative position of spatially-neighboring patches [48, 13]. The second uses external sensory information such as motor signals [2, 30] or sound [50, 3] to learn image transformations or categories. The third uses motion cues from videos [68, 31, 46, 51]. Although existing methods have demonstrated exciting results, these approaches often require delicate and cleverly-designed tasks in order to force the model to learn semantic features. Moreover, most existing methods learn only a *single task*. While the model could learn to perform really well at that task, it may in the process lose its focus on the actual intended task; i.e., to learn high-level semantic features. Recent self-supervised methods that do learn from multiple tasks either require a complex model to account for the potentially large differences in input data type (e.g., grayscale vs. color) and tasks (e.g., relative position vs. motion prediction) [14] or is designed specifically for tabletop robotic tasks and thus has difficulty generalizing to more complex real-world imagery [54].

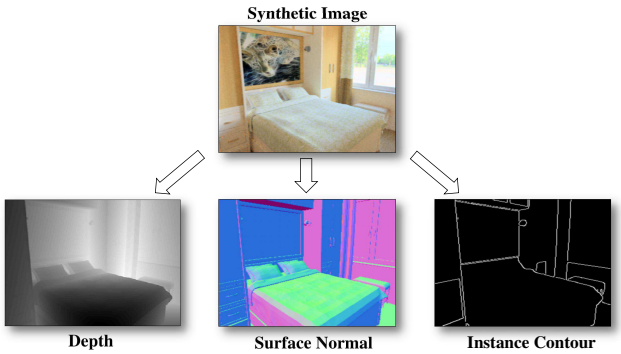

Figure 1. **Main idea.** A graphics engine can be used to easily render realistic synthetic images together with their various physical property maps. Using these images, we train a self-supervised visual representation learning algorithm in a multi-task setting that also adapts its features to real-world images.

In human learning, it is common to use multiple sources of information *jointly*. Babies explore a new object by looking at it, touching it, and even tasting it; humans learn a new language by listening, speaking, and writing in it. We

aim to use a similar strategy for visual representation learning. Specifically, by training a model to jointly learn several complementary tasks, we can force it to learn general features that are not overfit to a single task and are instead useful for a variety of tasks. However, multi-task learning using natural images would require access to different types of annotations (e.g., depth [16], surface normal [16, 45], segmentations [45]) for each image, which would be both expensive and time-consuming to collect.

Our main idea is to instead use *synthetic images* and their various *free* annotations for visual representation learning. Why synthetic data? First, computer graphics (CG) imagery is more realistic than ever and is only getting better over time. Second, rendering synthetic data at scale is easier and cheaper compared to collecting and annotating photos from the real-world. Third, a user has full control of a virtual world, including its objects, scenes, lighting, physics, etc. For example, the global illumination or weather condition of a scene can be changed trivially. This property would be very useful for learning a robust, invariant visual representation since the *same scene* can be altered in various ways *without changing the semantics*. Finally, the CG industry is huge and continuously growing, and its created content can often be useful for computer vision researchers. For example, [56] demonstrated how the GTA-V [1] game can be used to quickly generate semantic segmentation labels for training a supervised segmentation model.

Although synthetic data provides many advantages, it can be still challenging to learn general-purpose features applicable to real images. First, while synthetic images have become realistic, it's still not hard to differentiate them from real-world photos; i.e., there is a domain difference that must be overcome. To tackle this, we propose an unsupervised feature-level domain adaptation technique using adversarial training, which leads to better performance when the learned features are transferred to real-world tasks. Second, any semantic category label must still be provided by a human annotator, which would defeat the purpose of using synthetic data for self-supervised learning. Thus, we instead leverage other *free* physical cues to learn the visual representations. Specifically, we train a network that takes an image as input and predicts its depth, surface normal, and instance contour maps. We empirically show that learning to predict these mid-level cues forces the network to also learn transferable high-level semantics.

Contributions Our main contribution is a novel self-supervised multi-task feature learning network that learns from synthetic imagery while adapting its representation to real images via adversarial learning. We demonstrate through extensive experiments on ImageNet and PASCAL VOC that our multi-task approach produces visual representations that are better than alternative single-task baselines, and highly competitive with the state-of-the-art. We release our code and models on `jason718.github.io/project/cvpr18/main.html`

2. Related work

Synthetic data for vision CAD models have been used for various vision tasks such as 2D-3D alignment [6, 4], object detection [53], joint pose estimation and image-shape alignment [64, 27]. Popular datasets include the Princeton Shape Benchmark [60], ShapeNet [11], and SUNCG [63]. Synthetic data has also begun to show promising usage for vision tasks including learning optical flow [43], semantic segmentation [56, 57, 59], video analysis [20], stereo [75], navigation [80], and intuitive physics [39, 70, 47]. In contrast to these approaches, our work uses synthetic data to learn general-purpose visual representations in a self-supervised way.

Representation learning Representation learning has been a fundamental problem for years; see Bengio *et al*. [7] for a great survey. Classical methods such as the autoencoder [26, 67] learn compressed features while trying to recover the input image. Recent self-supervised approaches have shown promising results, and include recovering color from a grayscale image (and vice versa) [73, 74, 37], image inpainting [52], predicting the relative spatial location or equivariance relation of image patches [48, 13, 49], using motion cues in video [68, 31, 46, 51], and using GANs [15]. Other works leverage non-visual sensory data to predict egomotion between image pairs [2, 30] and sound from video [50, 3]. In contrast to the above works, we explore the advantage of using *multiple* tasks.

While a similar multi-task learning idea has been studied in [14, 54, 69], each have their drawbacks. In [14], four very different tasks are combined into one learning framework. However, because the tasks are very different in the required input data type and learning objectives, each task is learned one after the other rather than simultaneously and special care must be made to handle the different data types. In [54], a self-supervised robot learns to perform different tasks and in the process acquires useful visual features. However, it has limited transferability because the learning is specific to the tabletop robotic setting. Finally, [69] combines the tasks of spatial location prediction [13] and motion coherence [68], by first initializing with the weights learned on spatial location prediction and then continuing to learn via motion coherence (along with transitive relations acquired in the process). Compared to these methods, our model is relatively simple yet generalizes well, and learns all tasks simultaneously.

Domain adaptation To overcome dataset bias, *visual* domain adaptation was first introduced in [58]. Recent methods using deep networks align features by minimizing some distance function across the domains [65, 21]. GAN [25]

based pixel-level domain adaptation methods have also gained a lot of attention and include those that require paired data [29] as well as unpaired data [79, 32, 41].

Domain adaptation techniques have also been used to adapt models trained on synthetic data to real-world tasks [61, 9]. Our model also minimizes the domain gap between real and synthetic images, but we perform domain adaptation in feature space similar to [66, 22], whereby a domain discriminator learns to distinguish the domains while the learned representation (through a generator) tries to fool the discriminator. To our knowledge, our model is the first to adapt the features learned on synthetic data to real images for self-supervised feature learning.

Multi-task learning Multi-task learning [10] has been used for a variety of vision problems including surface normal and depth prediction [16, 17], semantic segmentation [45], pose estimation [24], robot manipulation [55, 54], and face detection [77]. Kokkinos [33] introduces a method to jointly learn low-, mid-, and high-level vision tasks in a unified architecture. Inspired by these works, we use multi-task learning for self-supervised feature learning. We demonstrate that our multi-task learning approach learns better representations compared to single-task learning.

3. Approach

We introduce our self-supervised deep network which jointly learns multiple tasks for visual representation learning, and the domain adaptor which minimizes the feature space domain gap between real and synthetic images. Our final learned features will be transferred to real-world tasks.

3.1. Multi-task feature learning

To learn general-purpose features that are useful for a variety of tasks, we train our network to simultaneously solve three different tasks. Specifically, our network takes as input a single synthetic image and computes its corresponding instance contour map, depth map, and surface normal map, as shown in Fig. 2.

Instance contour detection. We can easily extract instance-level segmentation masks from synthetic imagery. The masks are generated from pre-built 3D models, and are clean and accurate. However, the tags associated with an instance are typically noisy or inconsistent (e.g., two identical chairs from different synthetic scenes could be named 'chair1' and 'furniture2'). Fixing these errors (e.g., for semantic segmentation) would require a human annotator, which would defeat the purpose of self-supervised learning.

We therefore instead opt to extract edges from the instance-level segmentation masks, which alleviates the issues with noisy instance labels. For this, we simply run the canny edge detector on the segmentation masks. Since the edges are extracted from instance-level segmentations, they

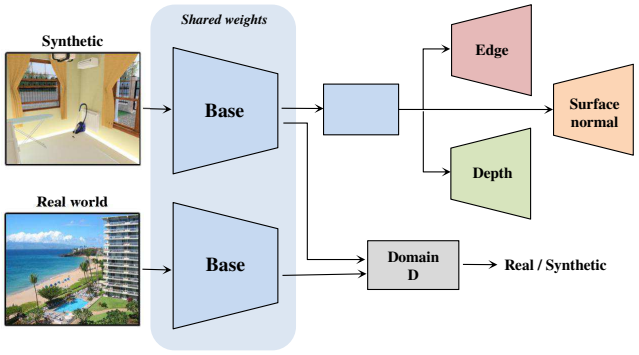

Figure 2. **Network architecture.** The upper net takes a synthetic image and predicts its depth, surface normal, and instance contour map. The bottom net extracts features from a real-world image. The domain discriminator D tries to differentiate real and synthetic features. The learned blue modules are used for transfer learning on real-world tasks.

correspond to *semantic* edges (i.e., contours of objects) as opposed to low-level edges. Fig. 1 shows an example; notice how the edges within an object, texture, and shadows are ignored. Using these semantic contour maps, we can train a model to ignore the low-level edges within an object and focus instead on the high-level edges that separate one object from another, which is exactly what we want in a high-level feature learning algorithm.

More specifically, we formulate the task as a binary semantic edge/non-edge prediction task, and use the class-balanced sigmoid cross entropy loss proposed in [71]:

$$L_e(E) = -\beta \sum_i \log P(y_i = 1|\theta) \; - (1-\beta) \sum_j \log P(y_j = 0|\theta)$$

where E is our predicted edge map, E' is the ground-truth edge map, $\beta = |E'_-|/|E'_- + E'_+|$, and $|E'_-|$ and $|E'_+|$ denote the number of ground-truth edges and non-edges, respectively, i indexes the ground-truth edge pixels, j indexes the ground-truth background pixels, θ denotes the network parameters, and $P(y_i = 1|\theta)$ and $P(y_j = 0|\theta)$ are the predicted probabilities for a pixel corresponding to an edge and background, respectively.

Depth prediction. Existing feature learning methods mainly focus on designing 'pre-text' tasks such as predicting the relative position of spatial patches [13, 48] or image in-painting [52]. The underlying physical properties of a scene like its depth or surface normal have been largely unexplored for learning representations. The only exception is the work of [5], which learns using surface normals corresponding to real-world images.[1]

Predicting the depth for each pixel in an image requires understanding high-level semantics about objects and their relative placements in a scene; it requires the model to figure out the objects that are closer/farther from the camera,

[1]Our multi-task AlexNet yields better transfer learning results on VOC07 detection than single-task VGG of [5]; 52.6% vs. 51.0% mAP.

and their shape and pose. While real-world depth imagery computed using a depth camera (e.g., the Kinect) can often be noisy, the depth map extracted from a synthetic scene is clean and accurate. To train the network to predict depth, we follow the approach of [17], which compares the predicted and ground-truth log depth maps of an image $Q = \log Y$ and $Q' = \log Y'$, where Y and Y' are the predicted and ground-truth depth maps, respectively. Their scale-invariant depth prediction loss is:

$$L_d(Q) = \frac{1}{n}\sum_i d_i^2 - \frac{1}{2n^2}\sum_{i,j} d_i d_j$$

where i indexes the pixels in an image, n is the total number of pixels, and $d = Q - Q'$ is the element-wise difference between the predicted and ground-truth log depth maps. The first term is the L2 difference and the second term tries to enforce errors to be consistent with one another in their sign.

Surface normal estimation. Surface normal is highly related to depth, and previous work [16, 17] show that combing the two tasks can help both. We use the inverse of the dot product between the ground-truth and the prediction as the loss [16]:

$$L_s(S) = -\frac{1}{n}\sum_i S_i \cdot S_i'$$

where i indexes the pixels in an image, n is the total number of pixels, S is the predicted surface normal map, and S' is the ground-truth surface normal map.

3.2. Unsupervised feature space domain adaptation

While the features learned above on multiple tasks will be more general-purpose than those learned on a single task, they will not be directly useful for real-world tasks due to the domain gap between synthetic and real images. Thus, we next describe how to adapt the features learned on synthetic images to real images.

Since our goal is to learn features in a self-supervised way, we cannot assume that we have access to any task labels for real images. We therefore formulate the problem as *unsupervised* domain adaptation, where the goal is to minimize the domain gap between synthetic $x_i \in X$ and real $y_j \in Y$ images. We follow a generative adversarial learning (GAN) [25] approach, which pits a generator and a discriminator against each other. In our case, the two networks learn from each other to minimize the domain difference between synthetic and real-world images so that the features learned on synthetic images can generalize to real-world images, similar to [22, 61, 9, 66]. Since the domain gap between our synthetic data and real images can be potentially huge (especially in terms of high-level semantics), we opt to perform the adaptation at the feature-level [22, 66] rather than at the pixel-level [61, 9].

Specifically, we update the discriminator and generator networks by alternating the following two stages. In the first stage, given a batch of synthetic images $\mathbf{x} = \{x_i\}$ and a batch of real images $\mathbf{y} = \{y_j\}$, the generator B (base network in Fig. 2) computes features $z_{x_i} = B(x_i)$ and $z_{y_j} = B(y_j)$ for each synthetic image x_i and real image y_j, respectively. The domain discriminator D then updates its parameters ϕ_D by minimizing the following binary cross-entropy loss:

$$L_D(\phi_D|z_\mathbf{x}, z_\mathbf{y}) = -\sum_i \log(D(z_{x_i})) - \sum_j \log(1 - D(z_{y_j}))$$

where we assign $1, 0$ labels to synthetic and real images x_i, y_j, respectively.

In the second stage, we fix D and update the generator B as well as the tasks heads H for the three tasks. Specifically, the parameters ϕ_B, ϕ_H are updated jointly using:

$$L_{BH}(\phi_B, \phi_H|z_\mathbf{x}) = -\sum_i \log(1 - D(z_{x_i})) + \lambda_e L_e(E_{x_i}) + \lambda_d L_d(Q_{x_i}) + \lambda_s L_s(S_{x_i}),$$

where $L_e(E_{x_i}), L_d(Q_{x_i}), L_s(S_{x_i})$ are the losses for instance contour, depth, and surface normal prediction for synthetic image x_i, respectively, and $\lambda_e, \lambda_d, \lambda_s$ are weights to scale their gradients to have similar magnitude. L_{BH} updates B so that D is fooled into thinking that the features extracted from a synthetic image are from a real image, while also updating H so that the features are good for instance contour, depth, and surface normal prediction.

Our training process is summarized in Alg. 1. Note that we do not directly update the generator B using any real images; instead the real images only directly update D, which in turn forces B to produce more domain-agnostic features for synthetic images. We also tried updating B with real images (by adding $-\sum_j \log(D(z_{y_j}))$ to L_{BH}), but this did not result in any improvement. Once training converges, we transfer B and finetune it on real-world tasks like ImageNet classification and PASCAL VOC detection.

Algorithm 1 Multi-task Adversarial Domain Adaptation

Input: Synthetic images X, real images Y, max iteration T
Output: Domain adapted base network B

for $t = 1$ to T **do**
  Sample a batch of synthetic images $\mathbf{x} = \{x_i\}$
  Sample a batch of real images $\mathbf{y} = \{y_j\}$
  Extract feature for each image: $z_{x_i} = B(x_i), z_{y_j} = B(y_j)$
  Keep D frozen, update B, H through $L_{BH}(\phi_B, \phi_H|z_\mathbf{x})$
  Keep B frozen, update D through $L_D(\phi_D|z_\mathbf{x}, z_\mathbf{y})$

3.3. Network architecture

Our network architecture is shown in Fig. 2. The blue base network consists of convolutional layers, followed by ReLU nonlinearity and BatchNorm [28]. The ensuing bottleneck layers (middle blue block) consist of dilated convolution layers [72] to enlarge the receptive field. In our experiments, the number of layers and filters in the base and bottleneck blocks follow the standard AlexNet [35] model to

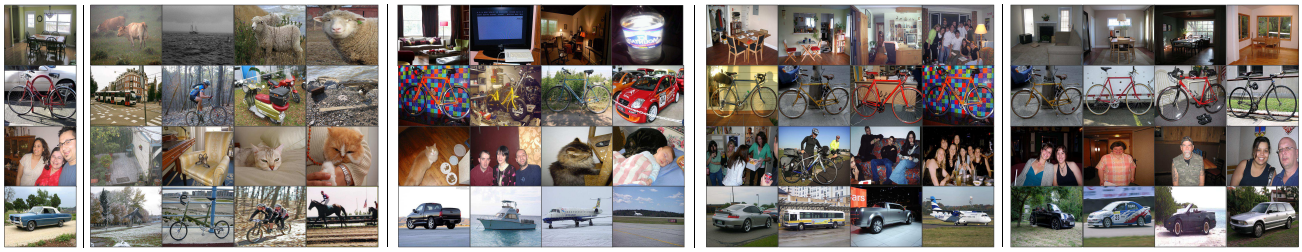

Figure 3. Nearest neighbor retrieval results. The first column contains the query images. We show the four nearest neighbors of a randomly initialized AlexNet, our model without and with domain adaptation, and ImageNet pre-trained AlexNet. See text for details.

ensure a fair comparison with existing self-supervised feature learning methods (e.g., [13, 73, 74, 49]). The task heads (red, green, and orange blocks) consist of deconvolution layers, followed by ReLU and BatchNorm [28]. Finally, the domain discriminator is a 13×13 patch discriminator [29], which takes 'conv5' features from the base network.

Empirically, we find that minimizing the domain shift in a mid-level feature space like 'conv5' rather than at a lower or higher feature space produces the best transfer learning results. In Sec. 4.4, we validate the effect of adaptation across different layers.

4. Results

In this section, we evaluate the quality and transferability of the features that our model learns from synthetic data. We first produce qualitative visualizations of our learned conv1 filters, nearest neighbors obtained using our learned features, and learned task predictions on synthetic data. We then evaluate on transfer learning benchmarks: fine-tuning the features on PASCAL VOC classification and detection, and freezing the features learned from synthetic data and then training a classifier on top of them for ImageNet classification. We then conduct ablation studies to analyze the different components of our algorithm. Finally, we evaluate our features on NYUD surface normal prediction.

4.1. Experimental setup

Architecture As described in Sec. 3.3, we set our base network to use the same convolutional and pooling layers as AlexNet [35] (the blue blocks in Fig. 2) to ensure a fair comparison with existing self-supervised approaches [73, 15, 13, 68, 31, 2, 50, 69]. We set our input to be grayscale by randomly duplicating one of the RGB channels three times since it can lead to more robust features [49, 13, 68].

Dataset We use Places365 [78] as the source of real images for domain adaptation, which contains 1.8 million images. For synthetic images, we combine SUNCG [63] and SceneNet RGB-D [44] to train our network. Both datasets come with depth maps for each synthetic image, and we compute instance contour maps from the provided instance masks. For surface normal, we use the ground-truth maps provided by [68] for SceneNet [44] and those provided by SUNCG [63].

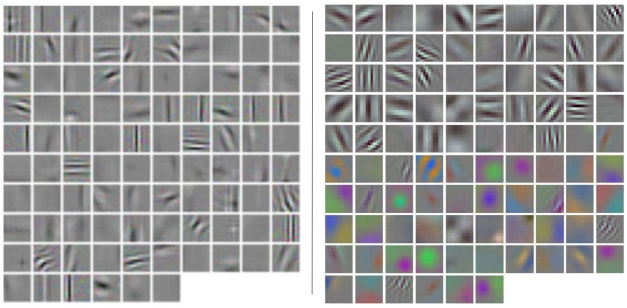

Figure 4. (**left**) The conv1 filters learned using our model on SUNCG and SceneNet. (**right**) The conv1 filters learned on ImageNet. While not as sharp as those learned on ImageNet, our model learns gabor-like conv1 filters.

4.2. Qualitative analysis without finetuning

Nearest neighbor retrieval We first perform nearest neighbor retrieval experiments on the PASCAL VOC 2012 trainval dataset. For this experiment, we compare a randomly initalized AlexNet, ImagenNet pretrained AlexNet, our model without domain adaptation, and our full model with domain adaptation. For each model, we extract conv5 features for each VOC image and retrieve the nearest neighbors for each query image.

Fig. 3 shows example results. We make several observations: (1) Both our full model and model without domain adaptation produces better features than randomly initialized features. (2) Since many of the ImageNet objects are not present in our synthetic dataset, our model is unable to distinguish between very similar categories but instead retrieves them together (e.g., cars, buses, and airplanes as the neighbor of query car). (3) Our full model performs better than our model without domain adaptation when there are humans or animals in the query images. This is likely because although these categories are never seen in our synthetic training set, they are common in Places [78] which we use for adaptation. (4) Compared to a pre-trained ImageNet [12] model, our full model is less discriminative and

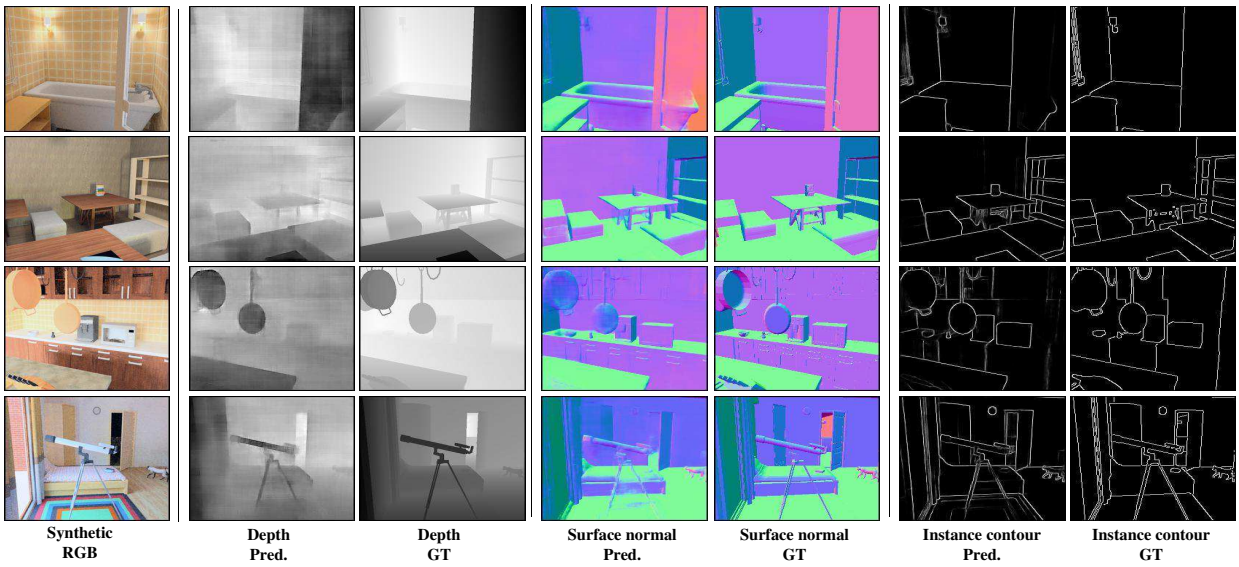

Figure 5. Representative examples of our model's depth, surface normal, and instance contour predictions on unseen SUNCG [63] images. Our network produces predictions that are sharp and detailed, and close to the ground-truth.

prefers to capture images with more objects in the image (e.g., third row with humans). This may again be due to Places [78] since it is a scene dataset rather than an object-centric dataset like ImageNet. Overall, this result can be seen as initial evidence that our pre-trained model can capture high-level semantics on real-world data.

Conv1 filter visualization In Fig. 4, we visualize the conv1 features learned on synthetic data. While not as sharp as those learned on ImageNet [12], our model learns conv1 features that resemble gabor-like filters. Since we always convert our input image to gray scale, our network does not learn any color blob filters.

Learned task prediction visualization We next show how well our model performs on the tasks that it is trained on. Fig. 5 shows our model's depth, surface normal, and instance contour predictions on unseen SUNCG [63] images. Overall, our predictions are sharp and clean, and look quite close to the ground-truth. Note that these are representative predictions and we only sampled these because they contain interesting failure cases. For example, in the first row there is a transparent glass door. Our network failures to capture the semantic meaning of a glass door and instead tries to predict the bathtub's surface normal and contours behind it. In the third row, our network fails to correctly predict the pan and pot's depth and surface normals due to ambiguity in 3D shape. This indicates that our network can struggle when predicting very detailed 3D properties. Similar results can been seen in the fourth row with the telescope body and legs. Finally, in the last row, there is a door whose inside is too dark to see. Therefore, our network predicts it as a wall but the ground-truth indicates there is actually something inside it.

These visualizations illustrate how well our network performs on each 'pre-text' task for feature learning. The better our model performs on these tasks, the better transferable features it is likely to get. In the remainder of the experiments, we demonstrate that this is indeed the case, and also provide quantitative evaluations on the surface normal 'pre-text' task in Sec. 4.5 where we fine-tune our network for surface normal estimation on NYUD [62].

4.3. Transfer learning

How well does our network generalize to new unseen data and tasks? To answer this, we perform experiments on various large-scale representation learning benchmarks.

Pascal VOC classification and detection We first evaluate on VOC classification following the protocol in [34]. We transfer the learned weights from our network (blue blocks Fig. 2) to a standard AlexNet [35] and then re-scale the weights using [34]. We then fine-tune our model's weights on VOC 2007 trainval and test on VOC 2007 test. Table 1 second column, shows the results. Our model outperforms all previous methods despite never having directly used any real images for pre-training (recall that the real images are only used for domain adaptation). In contrast, the existing methods are all trained on real images or videos. While previous research has mainly shown that synthetic data can be a good supplement to real-world imagery [56, 57], this result indicates the promise of directly using synthetic data and its free annotations for self-supervised representation learning.

We next test VOC detection accuracy using the Fast-RCNN [23] detector. We test two models: (1) finetuning on VOC 2007 trainval and testing on VOC 2007 test data; (2) finetuning on VOC 2012 train and testing on VOC 2012 val

Dataset Tasks	07 CLS	07 DET	12 DET
ImageNet [35]	79.9	56.8	56.5
Gaussian	53.4	41.3	-
Autoencoder [34]	53.8	41.9	-
Krahenbuel *et al.* [34]	56.6	45.6	42.8
Ego-equivariance [30]	-	41.7	-
Egomotion [2]	54.2	43.9	-
context-encoder [52]	56.5	44.5	
BiGAN [15]	58.6	46.2	44.9
sound [50]	61.3	-	42.9
flow [51]	61	52.2	48.6
motion [68]	63.1	47.2	43.5
clustering [8]	65.3	49.4	-
context [34]	65.3	51.1	49.9
colorization [73]	65.9	46.9	44.5
jigsaw [48]	67.6	**53.2**	-
splitbrain [74]	67.1	46.7	43.8
counting [49]	67.7	51.4	-
Ours	**68.0**	52.6	**50.0**

Table 1. Transfer learning results on PASCAL VOC 2007 classification and VOC 2007 and 2012 detection. We report the best numbers for each method reported in [34, 74, 49].

data. Table 1, right two columns show the results. Our models obtain the second best result on VOC 2007 and the best result on 2012. These results on detection verify that our learned features are robust and are able to generalize across different high-level tasks. More importantly, it again shows that despite using synthetic data (and real images only indirectly for domain adaptation), we can still learn transferable visual semantics.

ImageNet classification We next evaluate our learned features on ImageNet classification [12]. We freeze our network's pre-trained weights and train a multinomial logistic regression classifier on top of each layer from conv1 to conv5 using the ImageNet classification training data. Following [74], we bilinearly interpolate the feature maps of each layer so that the resulting flattened features across layers produce roughly equal number of dimensions.

Table 2 shows the results. Our model shows improvement over the different data initialization methods (Gaussian and Krähenbühl *et al.* [34]), but underperforms compared to the state-of-the-art. This is understandable since existing self-supervised approaches [13, 15, 52, 73] are trained on ImageNet, which here is also the test dataset. Our model is instead trained on synthetic indoor images, which can have quite different high-level semantics and thus has never seen most of the ImageNet categories during training (e.g., there are no dogs in SUNCG). Still, it outperforms [52] and performs similarly to [73] up through conv4, which shows that the learned semantics on synthetic data can still be useful for real-world image classification.

4.4. Ablation studies

We next perform ablation studies to dissect the contribution of the different components of our model. For this, we again use the PASCAL VOC classification and detection tasks for transfer learning.

method	conv1	conv2	conv3	conv4	conv5
ImageNet [35]	19.3	36.3	44.2	48.3	50.5
Gaussian	11.6	17.1	16.9	16.3	14.1
Krähenbühl *et al.* [34]	17.5	23.0	24.5	23.2	20.6
context [13]	16.2	23.3	30.2	31.7	29.6
BiGAN [15]	17.7	24.5	31.0	29.9	28.0
context-encoder [52]	14.1	20.7	21.0	19.8	15.5
colorization [73]	12.5	24.5	30.4	31.5	30.3
jigsaw [48]	**18.2**	28.8	34.0	33.9	27.1
splitbrain [74]	17.7	29.3	**35.4**	**35.2**	**32.8**
counting [49]	18.0	**30.6**	34.3	32.5	25.7
Ours	16.5	27.0	30.5	30.1	26.5

Table 2. Transfer learning results on ImageNet [12]. We freeze the weights of our model and train a linear classifier for ImageNet classification [12]. Our model is trained on synthetic data while all other methods are trained on ImageNet [12] (without labels). Despite the domain gap, our model still learns useful features for ImageNet classification.

Does multi-task learning help in learning semantics? We first analyze whether multi-task learning produces more transferable features compared to single-task learning. Table 3, first four rows show the transfer learning results of our final multi-task model ('3 tasks') versus each single-task model ('Edge', 'Depth', 'Surf.'). Our multi-task model outperforms all single-task models on both VOC classification and detection, which demonstrates that the tasks are complementary and that multi-task learning is beneficial for feature learning.

Does domain adaptation help? If so, on which layer should it be performed? Table 3, rows 5-8 show the transfer learning results after applying domain adaptation in different layers (i.e., in Fig. 2, which layer's features will go into the domain discriminator). We see that domain adaptation helps when performed on conv5 and conv6[2], which verifies that there is indeed a domain difference between our synthetic and real images that needs to be addressed. For example, on VOC classification, performing domain adaptation on conv5 results in 67.4% accuracy vs. 65.6% without domain adaptation. Interestingly, we see a slight decrease in performance from conv5 to conv6 across all tasks (rows 7 & 8). We hypothesize that this drop in performance is due to the biases in the synthetic and real-world image datasets we use: SUNCG and SceneNet are both comprised of indoor scenes mostly with man-made objects whereas Places is much more diverse and consists of indoor and outdoor scenes with man-made, natural, and living objects. Thus, the very high-level semantic differences may be hard to overcome, so domain adaptation can become difficult at the very high layers.

[2]Since our pre-text tasks are pixel prediction tasks, we convert fc6-7 of AlexNet into equivalent conv6-7 layers.

Task	Adaptation	#data	07-CLS	07-DET	12-DET
Edge	-	0.5M	63.9	46.9	44.8
Depth	-	0.5M	61.9	48.9	45.8
Surf.	-	0.5M	65.3	48.2	45.4
3 tasks	-	0.5M	**65.6**	**51.3**	**47.2**
3 tasks	conv1	0.5M	61.9	48.7	46
3 tasks	conv4	0.5M	63.4	49.5	46.3
3 tasks	conv5	0.5M	**67.4**	**52.0**	**49.2**
3 tasks	conv6	0.5M	66.9	51.5	48.2
3 tasks	conv5 Bi-fool	0.5M	66.2	51.3	48.5
3 tasks	conv5	1.5M	**68.0**	**52.6**	**50.0**

Table 3. Ablation studies. We evaluate the impact of multi-task learning, feature space domain adaptation, and amount of data on transfer learning. These factors contribute together to make our model learn transferable features from large-scale synthetic data.

We also see that it actually hurts to perform domain adaptation at a very low layer like conv1. The low performance on conv1 is likely due to the imperfect rendering quality of the synthetic data that we use. Many of the rendered images from SUNCG [63] are a bit noisy. Hence, if we take the first layer's conv1 features for domain adaptation, it is easy for the discriminator to overfit to this artifact. Indeed, we find that the conv1 filters learned in this setting are quite noisy, and this leads to lower transfer learning performance. By performing domain-adaptation at a higher level, we find that the competition between the discriminator and generator better levels-out, leading to improved transfer learning performance. Overall, performing domain adaptation in between the very low and very high layers, such as conv5, results in the best performance.

Does more data help? The main benefit of self-supervised or unsupervised learning methods is their scalability since they do not need any manually-labeled data. Thus, we next evaluate the impact that increasing data size has on feature learning. Specifically, we increase the size of our synthetic dataset from 0.5 million images to 1.5 million images. From Table 3, we can clearly see that having more data helps ('3task conv5' model, rows 7 vs. 10). Specifically, both classification and detection performance improve by 0.5-0.6% points.

Does fooling the discriminator both ways help? Since both of our real and synthetic images go through one base network, in contrast to standard GAN architectures, during the generator update we can fool the discriminator in both ways (i.e., generate synthetic features that look real and real image features that look synthetic). As seen in Table 3, row 9, fooling the discriminator in this way hurts the performance slightly, compared to only generating synthetic features that look real (row 7), but is still better than no domain adaptation (row 4). One likely reason for this is that updating the generator to fool the discriminator into thinking that a real image feature is synthetic does not directly help the generator produce good features for the synthetic depth, surface normal, and instance contour tasks (which are ultimately what is needed to learn semantics). Thus, by fooling the discriminator in both ways, the optimization process becomes unnecessarily tougher. This issue could potentially be solved using stabilizing methods such as a history buffer [61], which we leave for future study.

		Lower the better		Higher the better		
GT	Methods	Mean	Median	11.25°	22.5°	30°
[16]	Zhang *et al*. [76]	22.1	14.8	**39.6**	65.6	75.3
[16]	Ours	**21.9**	**14.6**	39.5	**66.7**	**76.5**
[36]	Wang *et al*. [69]	26.0	18.0	33.9	57.6	67.5
[36]	Ours	**23.8**	**16.2**	**36.6**	**62.0**	**72.9**

Table 4. Surface normal estimation on the NYUD [62] test set.

4.5. Surface normal on NYUD

Finally, we evaluate our model's transfer learning performance on the NYUD [62] dataset for surface normal estimation. Since one of our pre-training tasks is surface normal estimation, this experiment also allows us to measure how well our model does in learning that task. We use the standard split of 795 images for training and 654 images for testing. The evaluation metrics we use are the **Mean, Median, RMSE** error and percentage of pixels that have angle error less than **11.25°, 22.5°, and 30°** between the model predictions and the ground-truth predictions. We use both the ground-truths provided by [36] and [16].

We compare our model with the self-supervised model of [69], which pre-trains on the combined tasks of spatial location prediction [13] and motion coherence [68], and the supervised model trained with synthetic data [76], which pre-trains on ImageNet classification and SUNCG surface normal estimation. For this experiment, we use an FCN [42] architecture with skip connections similar to [76] and pre-train on 0.5 million SUNCG synthetic images on joint surface normal, depth, and instance contour prediction.

Table 4 shows the results. Our model clearly outperforms [69], which is somewhat expected since we directly pre-train on surface normal estimation as one of the tasks, and performs slightly better than [76] on average. Our model still needs to adapt from synthetic to real images, so our good performance likely indicates that (1) our model performs well on the pre-training tasks (surface normal estimation being one of them) and (2) our domain adaptation reduces the domain gap between synthetic and real images to ease fine-tuning.

5. Conclusion

While synthetic data has become more realistic than ever before, prior work has not explored learning general-purpose visual representations from them. Our novel cross-domain multi-task feature learning network takes a promising step in this direction.

Acknowledgements. This work was supported in part by NSF under Grant No. 1748387, AWS Cloud Credits for Research Program, and GPUs donated by NVIDIA.

References

[1] Grand theft auto five(v). www.rockstargames.com/V/. 2

[2] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *ICCV*, 2015. 1, 2, 5, 7

[3] R. Arandjelovic and A. Zisserman. Look, listen and learn. *ICCV*, 2017. 1, 2

[4] M. Aubry, D. Maturana, A. Efros, B. Russell, and J. Sivic. Seeing 3d chairs: exemplar part-based 2d-3d alignment using a large dataset of cad models. In *CVPR*, 2014. 2

[5] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan. Pixelnet: Representation of the pixels, by the pixels, and for the pixels. In *arXiv:1702.06506*, 2017. 3

[6] A. Bansal, B. Russell, and A. Gupta. Marr revisited: 2d-3d model alignment via surface normal prediction. In *CVPR*, 2016. 2

[7] Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. In *PAMI*, 2013. 2

[8] P. Bojanowski and A. Joulin. Unsupervised learning by predicting noise. In *ICML*, 2017. 7

[9] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *CVPR*, 2017. 3, 4

[10] R. Caruana. Multitask learning. *Machine Learning*, 1997. 3

[11] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu. Shapenet: An information-rich 3d model repository. In *arXiv:1512.03012*, 2015. 2

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009. 1, 5, 6, 7

[13] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 1, 2, 3, 5, 7, 8

[14] C. Doersch and A. Zisserman. Multi-task self-supervised visual learning. In *ICCV*, 2017. 1, 2

[15] J. Donahue, P. Krähenbühl, and T. Darrell. Adversarial feature learning. In *ICLR*, 2017. 2, 5, 7

[16] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *ICCV*, 2015. 2, 3, 4, 8

[17] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *NIPS*, 2014. 3, 4

[18] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. In *IJCV*, 2010. 1

[19] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: an incremental bayesian approach tested on 101 object categories. In *CVPR Workshop*, 2004. 1

[20] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *CVPR*, 2016. 2

[21] Y. Ganin and V. S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015. 2

[22] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016. 3, 4

[23] R. B. Girshick. Fast R-CNN. In *ICCV*, 2015. 1, 6

[24] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik. R-cnns for pose estimation and action detection. In *arXiv*, 2014. 3

[25] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014. 2, 4

[26] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. In *Science*, 2006. 1, 2

[27] Q. Huang, H. Wang, and V. Koltun. Single-view reconstruction via joint analysis of image and shape collections. *SIGGRAPH*, 2015. 2

[28] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICML*, 2015. 4, 5

[29] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017. 3, 5

[30] D. Jayaraman and K. Grauman. Learning image representations tied to egomotion. In *ICCV*, 2015. 1, 2, 7

[31] D. Jayaraman and K. Grauman. Slow and steady feature analysis: higher order temporal coherence in video. In *CVPR*, 2016. 1, 2, 5

[32] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. *ICML*, 2017. 3

[33] I. Kokkinos. Ubernet: Training a 'universal' convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. *CVPR*, 2017. 3

[34] P. Krähenbühl, C. Doersch, J. Donahue, and T. Darrell. Data-dependent initializations of convolutional neural networks. In *ICLR*, 2016. 6, 7

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 4, 5, 6, 7

[36] L. Ladicky, B. Zeisl, and M. Pollefeys. Discriminatively trained dense surface normal estimation. In *ECCV*, 2014. 8

[37] G. Larsson, M. Maire, and G. Shakhnarovich. Colorization as a proxy task for visual understanding. *CVPR*, 2017. 2

[38] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998. 1

[39] A. Lerer, S. Gross, and R. Fergus. Learning physical intuition of block towers by example. In *ICML*, 2016. 2

[40] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *CoRR*, 2014. 1

[41] M. Liu and O. Tuzel. Coupled generative adversarial networks. *NIPS*, 2016. 3

[42] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1, 8

[43] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *CVPR*, 2016. 2

[44] J. McCormac, A. Handa, S. Leutenegger, and A. J.Davison. Scenenet rgb-d: Can 5m synthetic images beat generic imagenet pre-training on indoor segmentation? In *ICCV*, 2017. 5

[45] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert. Cross-stitch networks for multi-task learning. In *CVPR*, 2016. 2, 3

[46] I. Misra, C. L. Zitnick, and M. Hebert. Shuffle and learn: Unsupervised learning using temporal order verification. In *ECCV*, 2016. 1, 2

[47] R. Mottaghi, H. Bagherinezhad, M. Rastegari, and A. Farhadi. Newtonian image understanding: Unfolding the dynamics of objects in static images. In *CVPR*, 2016. 2

[48] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *ECCV*, 2016. 1, 2, 3, 7

[49] M. Noroozi, H. Pirsiavash, and P. Favaro. Representation learning by learning to count. In *ICCV*, 2017. 1, 2, 5, 7

[50] A. Owens, J. Wu, J. McDermott, W. Freeman, and A. Torralba. Ambient sound provides supervision for visual learning. In *ECCV*, 2016. 1, 2, 5, 7

[51] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. In *CVPR*, 2017. 1, 2, 7

[52] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 2, 3, 7

[53] X. Peng, B. Sun, K. Ali, and K. Saenko. Learning deep object detectors from 3d models. In *ICCV*, 2015. 2

[54] L. Pinto, D. Gandhi, Y. Han, Y. Park, and A. Gupta. The curious robot: Learning visual representations via physical interactions. In *ECCV*, 2016. 1, 2, 3

[55] L. Pinto and A. Gupta. Learning to push by grasping: Using multiple tasks for effective learning. In *ICRA*, 2017. 3

[56] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 2, 6

[57] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. Lopez. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 2, 6

[58] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *ECCV*, 2010. 2

[59] A. Shafaei, J. J. Little, and M. Schmidt. Play and learn: Using video games to train computer vision models. In *BMVC*, 2016. 2

[60] P. Shilane, P. Min, M. Kazhdan, and T. Funkhouser. The Princeton shape benchmark. In *Shape Modeling International*, 2004. 2

[61] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. *CVPR*, 2017. 3, 4, 8

[62] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 6, 8

[63] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. Semantic scene completion from a single depth image. *CVPR*, 2017. 2, 5, 6, 8

[64] H. Su, Q. Huang, N. J. Mitra, Y. Li, and L. Guibas. Estimating image depth using shape collections. *SIGGRAPH*, 2014. 2

[65] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, 2015. 2

[66] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 3, 4

[67] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. In *JMLR*, 2010. 1, 2

[68] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *ICCV*, 2015. 1, 2, 5, 7, 8

[69] X. Wang, K. He, and A. Gupta. Transitive invariance for self-supervised visual representation learning. In *ICCV*, 2017. 2, 5, 8

[70] J. Wu, I. Yildirim, J. J. Lim, W. T. Freeman, and J. B. Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. In *NIPS*, 2015. 2

[71] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, 2015. 1, 3

[72] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2015. 4

[73] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *ECCV*, 2016. 1, 2, 5, 7

[74] R. Zhang, P. Isola, and A. A. Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *CVPR*, 2017. 1, 2, 5, 7

[75] Y. Zhang, W. Qiu, Q. Chen, X. Hu, and A. L. Yuille. Unrealstereo: A synthetic dataset for analyzing stereo vision. In *arXiv*, 2016. 2

[76] Y. Zhang, S. Song, E. Yumer, M. Savva, J.-Y. Lee, H. Jin, and T. Funkhouser. Physically-based rendering for indoor scene understanding using convolutional neural networks. *CVPR*, 2017. 8

[77] Z. Zhang, P. Luo, C. C. Loy, and X. Tang. Facial landmark detection by deep multi-task learning. In *ECCV*, 2014. 3

[78] B. Zhou, A. Khosla, A. Lapedriza, A. Torralba, and A. Oliva. Places: An image database for deep scene understanding. In *arXiv*, 2016. 5, 6

[79] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 3

[80] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *ICRA*, 2017. 2