Taskonomy: Disentangling Task Transfer Learning
Supplementary Material

Amir R. Zamir[1,2] Sasha Sax[1*] William Shen[1*] Leonidas Guibas[1] Jitendra Malik[2] Silvio Savarese[1]

[1] Stanford University [2] University of California, Berkeley

http://taskonomy.vision/

Abstract

The following items are provided in the supplementary material:

1. *Higher-order transfer experiments.*
2. *A tiny-data taxonomy.*
3. *Optimal transfer policies for selected tasks.*
4. *Our networks as feature extractors on MIT Places.*
5. *BIP Solver as API.*
6. *Significance testing of taxonomy.*
7. *Qualitative results of network output.*
8. *Agglomerative clustering of tasks.*
9. *Analysis of transitive transfers.*
10. *Experimental setup details.*
11. *Task dictionary definition and discussion.*

1. Higher-Order Transfers

In the main paper, low-order transfers (orders 2-4) often achieved superior performance to first-order transfers. What happens as we continue to increase the order? Do we get the best performance from using the entire bank of task-specific networks?

In addition to training many first-order and low-order (orders 2-4) transfers, we trained one transfer function for each order 5-25. Each high-order transfer function was trained on the combined representations of the top k first-order transfers. The high-order transfers are available in the provided API. The calculated distances matrix including higher-order transfers is shown in figure 2.

2. Tiny Data Taxonomy

As the amount of supervision of the transfer networks increases, the performance margin between them decreases. Eventually, full supervision is possible. Clearly, the amount of data available for training transfers is an important factor.

Figure 1: **Selected policies.** For tasks in the table form Figure 10 of the main paper, we show the best policies for transferring to that task. The center node of each cluster is the target task, and the surrounding nodes at the tail of each arrow are the suggested source tasks.

We present a low-data taxonomy experiment in which the transfer networks were trained on 1k data points, as opposed to the main experiments which used 16k data points. The distance matrix for these transfers is in figure 3, and the transfers themselves are accessible in the provided API.

*equal.

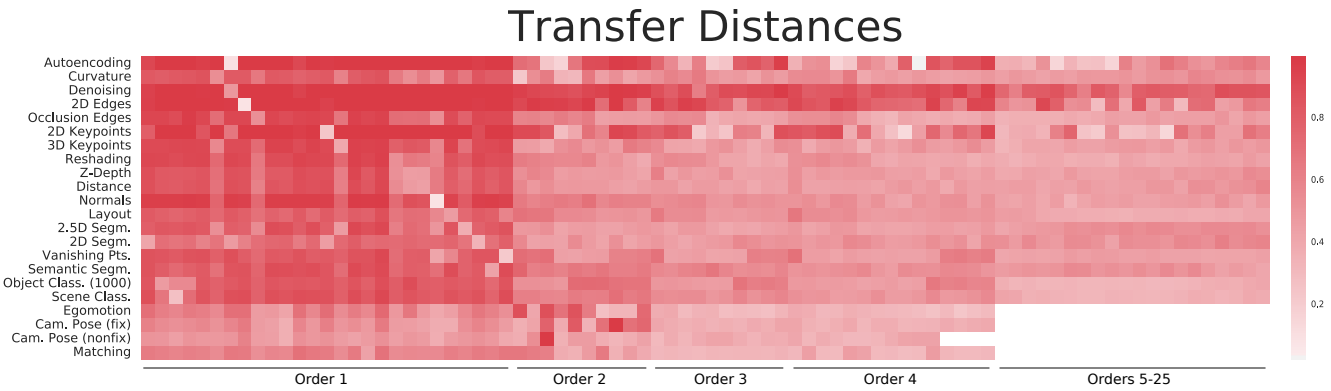


Figure 2: **Higher Order Distances.** Calculated task distances for all 16k transfers. The columns are sorted by transfer order and the orders are shown along the x-axis. Each row is a target task.

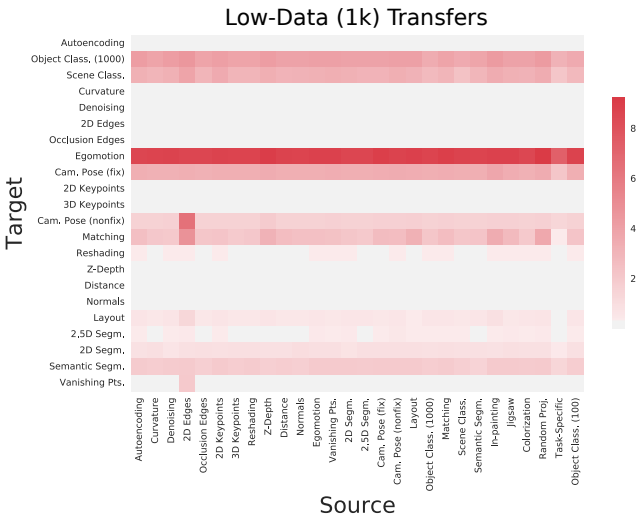


Figure 3: **Low data transfers.** Expected loss after training transfer functions with 1k samples (shown before AHP normalization). This is a low-data version of the transfer loss matrix in the main paper.

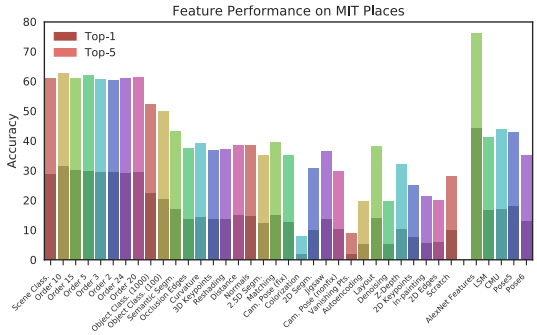


Figure 4: **Feature extraction.** Performance of readout functions trained on MIT Places. We compare our networks to common baselines. The x-axis is ordered according to transfer performance on our dataset, and the y-axis shows Top-1 and Top-5 accuracy on MIT Places. The sources on the left were trained on our dataset, while the right portion contains external baselines.

3. Special Task Policies

In the main paper, we showed the best transfer performance of our networks against some commonly used baselines. In figure 1, we now show the collections of transfers that beat the baselines shown in figure 9 in the main paper. In some cases, higher-order transfers outperformed the ones in figure 9, and we show those instead.

4. Feature Performance on MIT Places

When data labels are scarce, AlexNet and other pre-trained networks are commonly used as feature extractors that provide input for a small readout function. We compare the performance of our task-specific features to AlexNet features on 16k images from the 63 selected classes on the MIT Places dataset. While AlexNet was trained on 1M hand-labeled images, our task-specific networks were trained on 120k images where labels were generated automatically. The final results of the MIT feature extraction experiment are shown in figure 4.

5. BIP Solver as API

The BIP solver allows users to query our results and quickly compute the best transfer taxonomy for their particular use case. In this sense it can acts as an API, and we refer to it as the Transfer Learning API, or just the API. http://taskonomy.vision/api provides access to this API via a webpage and allows interactively entering the arguments and computing the corresponding transfer taxonomy. Qualitative transfer results corresponding to the computed taxonomy exemplified over a YouTube video is also shown.

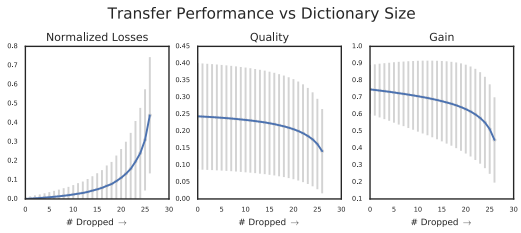


Figure 5: **Drop-k performance.** Expected performance of transferring to a task after dropping k tasks from our dictionary. The grey bars show the standard deviation of the performance.

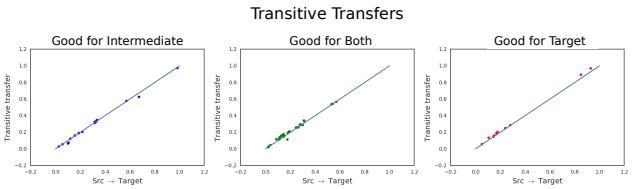


Figure 6: **Transitivity.** Loss after transitive transfer is on the y-axis and loss after direct ($A \rightarrow C$) transfer is on the x-axis. The line x=y is shown. Red dots are transfers which transferred well to C but not to B in first-order. Blue dots transferred well to the target, C, but not to B. Green dots transferred well to both.

6. Sensitivity to Choice of Dictionary

How universal are our findings with respect to the modeling dictionary? Is our coverage of the task space sufficiently dense? We evaluate this by randomly dropping some tasks from the dictionary and evaluating first-order transfer performance to tasks, and averaging over tasks. If our dictionary is dense, we should see a gradual drop in performance until our dictionary becomes too sparse, at which point performance will degrade rapidly. Results are shown in figure 5.

Additionally, in order to give a more comprehensive idea of the opportunity costs of choosing sub-optimal transfers, we show the losses, quality, and gain of all first-order transfers in figure 8.

7. Transfer via Transitivity

Perhaps transfer can be done in a transitive manner: first learn task A from images, and then B from A, and finally C from B. Intuitively, the performance $A \rightarrow C$ might improve if existing neural network machinery can take advantage of the additional computational power and supervision in the transitive $A \rightarrow B \rightarrow C$.

See figure 6 for the a comparison of the two transfer types. Based on the intermediate transfer B, we show three cases for transitive transfers: B transferred well to only A, only C, or both A and C. We found that the loss for transfers $A \rightarrow B \rightarrow C$ is very close to the loss for $A \rightarrow C$. This means that the largest contributor to performance on C is the representations learned for A, and that neither a doubling of the transfer computation nor the additional supervision for a proxy task has a strong effect on the performance. Since these results suggest that NNs are unable to effectively use transitive transfers, even if we modeled them, we are enforcing the taxonomy to be free of chains longer than length 1.

8. I. Model Specificness.

In principle, the findings are model dependent as a natural consequence of employing a fully computational formulation. However, we performed a set of experiments with varying transfer network and task-specific network architectures to quantify model independence within the function class of neural nets. The AHP affinity matrices turned out to be very similar (left) and the variance matrices (right) show pretty low values, signifying a high invariance.

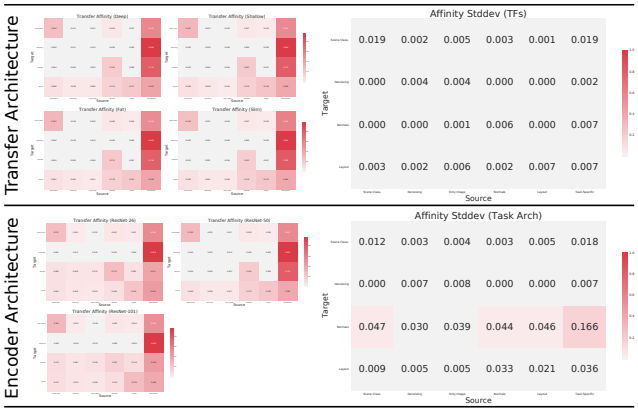


Figure 7: **Model Specificity.** (left) Affinity matrices. (right) Standard Deviation of Affinities.

9. Comparison to per-task state-of-the-art methods

We reported the win rate of our task specific models vs. two statistical baselines of average estimator *avg* (the best statistically informed guess) and a supervised readout network trained on a random (Gaussian) network's representation (*rand*) in the main paper. To get a sense of the quality of our networks vs. state-of-the-art task-specific methods, we compared our depth estimator vs. released models of Fully Convolutional Residual Network [19]. The results are provided in the table below showing our task specific network outperformed [19]. In general, we found the task-specific networks to perform on par or better than state-of-the-art for many of the tasks, though we do not formally benchmark or claim this.

Evaluation Pair	Win Rate	Loss
Ours (120k training images) v FCRN	80%	**0.39** v 0.47
Ours (3M training images) v FCRN	88%	**0.35** v 0.47

10. Evaluation on MIT Places & ImageNet: Experimental Details

Does our ranking of transfers hold across datasets? Is the taxonomy ordering something innate to the tasks, or an artifact of our data? We evaluated this on MIT Places and ImageNet, and found a very strong correlation between our ordering and the one on the new dataset; the results are reported in the main paper. The setup for these experiments is as follows:

Transfer to MIT Places: We evaluated the order on Places-365 in the following way. For each task in our dataset, we reinitialized a new transfer function, attached it to the task-specific encoder, and trained the network normally on 16k images from the 63 classes in the Home+Hotel and Work subgroups from Places-365, classes which were likely to be present in our dataset.

Note that the places training regimen is different from the one used to form the taxonomy, and the training and evaluation occur on different datasets. Despite this, correlation between the two orders was very high.

Transfer to ImageNet: Similar to our Places experiment, we selected a subset of categories likely to appear in our dataset (e.g. appliances, furniture). This had little effect on the ordering on our dataset. Then we distilled a trained ResNet-152 [11] and train a transfer function to predict the activations.

11. Baselines

To evaluate the performance of our networks, we compare our performance against 6 common baselines. Four of the baselines are trained on different external datasets:

1. **ImageNet fc7[18]**, with representation size 4096, is trained on ImageNet [33] 1000l way classification.
2. **Wang et al. [42]**, with representation size 1024, is trained on 100K videos from YouTube for unsupervised patch triplet ranking.
3. **Agrawal et al. [2]**, with representation shape $13 \times 13 \times 256$, is an unsupervised method trained on KITTI to do egomotion in outdoor environments.
4. **Zamir et al. [48]**, with representation size 500, is trained to do both relative camera pose prediction and point matching.

Two of the baselines are trained as standard tasks on our dataset: Colorization [49] and Context Prediction (Jigsaw) [29], and details are defined in sec. 14.

In the comparison to baseline tasks shown in Figure 10 of the main paper, the experimental setup was as follows: we extracted features for each baseline network, and then trained transfer functions to predict the labels of other tasks using the extracted representations. In the case where the baseline features were not of size 2048, we added an extra fully-connected layer. We then evaluated the quality and gain of our networks over these baselines using our standard ordinal metric.

We also compare our networks to these baselines on other datasets for which our networks were not trained. We compare the features extracted by our networks to those extracted by the baselines on MIT Places [51] by training readout functions. Results are shown in 4.

12. Network Architecture

We used Tensorflow for our implementation.

Encoder: For all tasks we modified a ResNet-50 encoder with no average-pooling and replace the last stride 2 convolution with stride 1. This gives us an output shape of $16 \times 16 \times 2048$; we use a 3×3 convolution to transform the output to our final representation, which is of shape $16 \times 16 \times 8$.

Decoder: The shape of the decoder depends on the task. For pixels-to-pixels prediction, we used a 15-layer fully convolutional decoder: 5 convolutional layers, and then alternating convolution and convolution_transpose layers. For low-dimensional tasks such as Vanishing Points, our decoder was two or three fully-connected layers with dropout applied after all but the last one. For all layers (except output), we used batchnorm.

Transfer Function: Our transfer functions have a tower structure, and multiple towers have their output concatenated channelwise. Our towers are feedforward with the following set of functions: first clip extreme values to 5-sigma and then do batch renormalization, followed by two convolutional layers. In the case where the readout function takes an image as input, these layers are stride-4 dilated convolutions. As our transfer functions may use both an image and a representation, we used two towers in every transfer function – if there was no input image, we fed a representation to both towers.

Loss Functions: For pixels-to-pixels prediction tasks we used a CGAN in addition to the standard loss function. The GAN training began after 25000 steps, and the discriminator was 5 layers with $10\times$ the standard weight decay. The loss was a linear combination of the $0.996\times$ the standard loss and $0.004\times$ the GAN loss. We found that including even this small amount of GAN loss helped to sharpen the images.

Hyperparameters: We used ADAM with a learning rate of 10^{-4} and annealed the learning rate by a factor of 10 after 80000 steps. We used batch size 32 and weight decay

2×10^{-6}.

Data: Out of more than 1M images collected, we used 120k from 87 buildings to train task-specific networks. The transfer networks were trained on 16k images from a separate 24 buildings. The taxonomy was generated from these transfer networks on a third set of 17k images from 13 buildings.

13. Pseudo-semantics Annotations

As we do not have semantic annotations on our dataset, we gathered pseudo-semantic annotations by using knowledge distillation [13] approach. That is, we labeled our dataset using the output of state-of-the-art large fully-supervised network (ResNet-151) for semantic objects (using ImageNet, 100 applicable classes), scene categories (using MIT Places [51], 63 indoor workplace and home classes in MIT Places scene hierarchy), and semantic object segmentation (using COCO [21], 17 applicable classes). In a user study, human annotators indicated that only 6.3% of images have no plausible label in their top-5 classes in Scene categories, showing that such labels are reliable enough and effective for being a gateway to modeling semantics in task space. The quality of the semantic labels can be seen in the provided sample of qualitative results or in the video. The list of selected classes for COCO and ImageNet are available here.

14. Detailed Definition of Task Dictionary

Task definitions are presented in alphabetical order:

Autoencoding PCA is a widely used method of understanding data by finding a low-dimensional latent representation. Autoencoding [14] is a nonlinear generalization of PCA that was originally proposed with transfer learning in mind: better downstream performance through autoencoding pretraining.

Colorization [49] Colorization requires taking a grayscale image and predicting the original colors. It is an unsupervised task, but also one that is semantic-aware[6]. For example, predicting the color of a fruit is simple once the fruit is identified.

Context Encoding Context Encoding was first introduced by Pathak et al. [30] and is a version of autoencoding where a large portion of the input is masked from the model. In order to fill in the occluded area, the model must reason about the scene geometry and semantics. Similar to colorization, it is an unsupervised-yet-still-semantic task.

Content Prediction (Jigsaw)[29] A discriminative version of Context Encoding, Jigsaw [29] requires a network to unscramble a permuted tiling of the input image.

Curvature Estimation Curvature-based features are excellent for identification because they are invariant under rigid transformations. Curvature is known to be important in visual processing–so much so that the Macaque visual cortex has a dedicated curvature processing region [46].

Denoising It is desirable for similar inputs to have similar representations, but representations learned by autoencoding are excessively sensitive to perturbations in the input. Denoising [40] (autoencoding) encourages limited invariance by mapping slightly perturbed inputs to the unperturbed input.

Depth Estimation, Euclidean Depth estimation is an important task, useful for detecting proximity to obstacles and items of interest. It is also a useful intermediate step for agents to localize themselves in 3D space. Euclidean depth is measured as the distance from each pixel to the camera's optical center.

Depth Estimation, Z-Buffer As opposed to Euclidean depth estimation, researchers typically use z-buffer depth which is defined as the distance to the camera plane. This is not the way that humans typically perceive depth, but is included because this is the standard formulation and all of our depth-derived tasks are derived from *z-buffer*.

Edge Detection (2D) Edge detection is historically a fundamental task in computer vision. Edges are commonly used as an intermediate representation or as a feature in a larger processing pipeline. We include the output of a Canny edge detector without nonmax suppression (to make the task learnable by neural networks).

Edge Detection (3D) As opposed to 2D edges, we define 3D edges as "occlusion edges," or edges where an object in the foreground obscures something behind it. 2D edges respond to changes in texture, but 3D edges are features which depend only on the 3D geometry and are invariant to color and lighting.

Keypoint Detection (2D) Keypoint detection has a long history in computer vision, and is useful for many, many tasks. Keypoint algorithmns usually consist of two parts, both a keypoint detector and some local patch descriptor which is invariant across multiple images [24, 4, 32]. 2D keypoint detection encourages the network to identify locally important regions of an image, and *point matching* encourages the network to learn feature descriptors. Identifying keypoints is frequently still a first step in a larger visual pipeline. We use the output of SURF [4] (before nonmax suppression) as our ground-truth.

Keypoint Detection (3D) 3D keypoints are similar to 2D keypoints except that they are derived from 3D data and therefore account for scene geometry. They are often invariant to informative (but possibly distracting) features such as textures [50, 39, 25, 47, 16]. We use the output of the NARF [39] algorithm (before nonmax suppression) as our 3D keypoint ground-truth.

Point Matching Deep networks trained for point matching learn feature descriptors that prove useful for downstream tasks. Point matching has applications in fine-

grained classification [44] and object recognition [23], multi-view reconstruction [36] and structure from motion [27], wide baseline matching [41], SLAM [35] and visual odometry[52].

Relative Camera Pose Estimation, Non-Fixated The famous "Kitten Carousel" experiment by Held and Hein [12] suggested that taking action is crucial for strong perception. Although more recent works call the original conclusion into question [31], the ability to localize oneself remains important for locomotion. For two different views with the same optical centers, we try to predict the 6-DOF relative camera pose (yaw, pitch, roll, xyz translation) between them.

Relative Camera Pose Estimation, Fixated We also include a simpler variant of camera pose estimation for which the center pixel of the two inputs is always the same physical 3D point. This problem is simpler in the sense that there are only five degrees of freedom.

Relative Camera Pose Estimation, Triplets (Egomotion) Videos are a common object of study in computer vision (e.g. visual odometry [9, 28]) and they provide dense data with high redundancy. We therefore include camera pose matching for input triplets with a fixed center point. With three images, models have a greater ability to match points for accurate localization.

Reshading One way to infer scene geometry is "shape from shading" [3] using the intrinsic image decomposition $I = A \cdot S$, where S is a shading function parameterized by lighting and depth. This decomposition is thought to be useful in human visual perception [1]. We define reshading as follows: Given an RGB image, the label is the shading function S that results from having a single point light at the camera origin, and S is multiplied by a constant fixed albedo.

Room Layout Estimation Estimating and aligning a 3D bounding box is a mid-level task that includes vanishing point estimation as a sub-problem, and has applications for robotic navigation [38], scene reconstruction [15], and augmented reality [8]. A variant of room layout estimation was used in the LSUN room layout challenge [45], but that formulation is ill-posed when there is camera roll or when no room corners are in view. Instead, we offer a formulation that remains well-defined regardless of the camera pose and field of view. This task includes some semantic information such as 'what constitutes a room' while simultaneously also including scene geometry.

Segmentation, Unsupervised (2D)) Gestalt psychologists proposed principles of grouping as a mechanism through which humans learn to perceive the world as a set of coherent objects [43]. Normalized cuts [37] are one method for segmenting images into perceptually similar groups, and we include this Gestalt task in our dictionary.

Segmentation, Unsupervised (2.5D)) Segmentation 2.5D uses the same algorithm as 2D, but the labels are computed jointly from the RGB image, the aligned depth image, and the aligned surface normals image. Therefore the 2.5D segmentation applies the principles not just to the world as it seems (in the RGB image), but also to the world as it is (ground-truth 3D). 2.5D segmentation incorporates information about the scene geometry that is not directly present in the RGB image but that is readily inferred by humans.

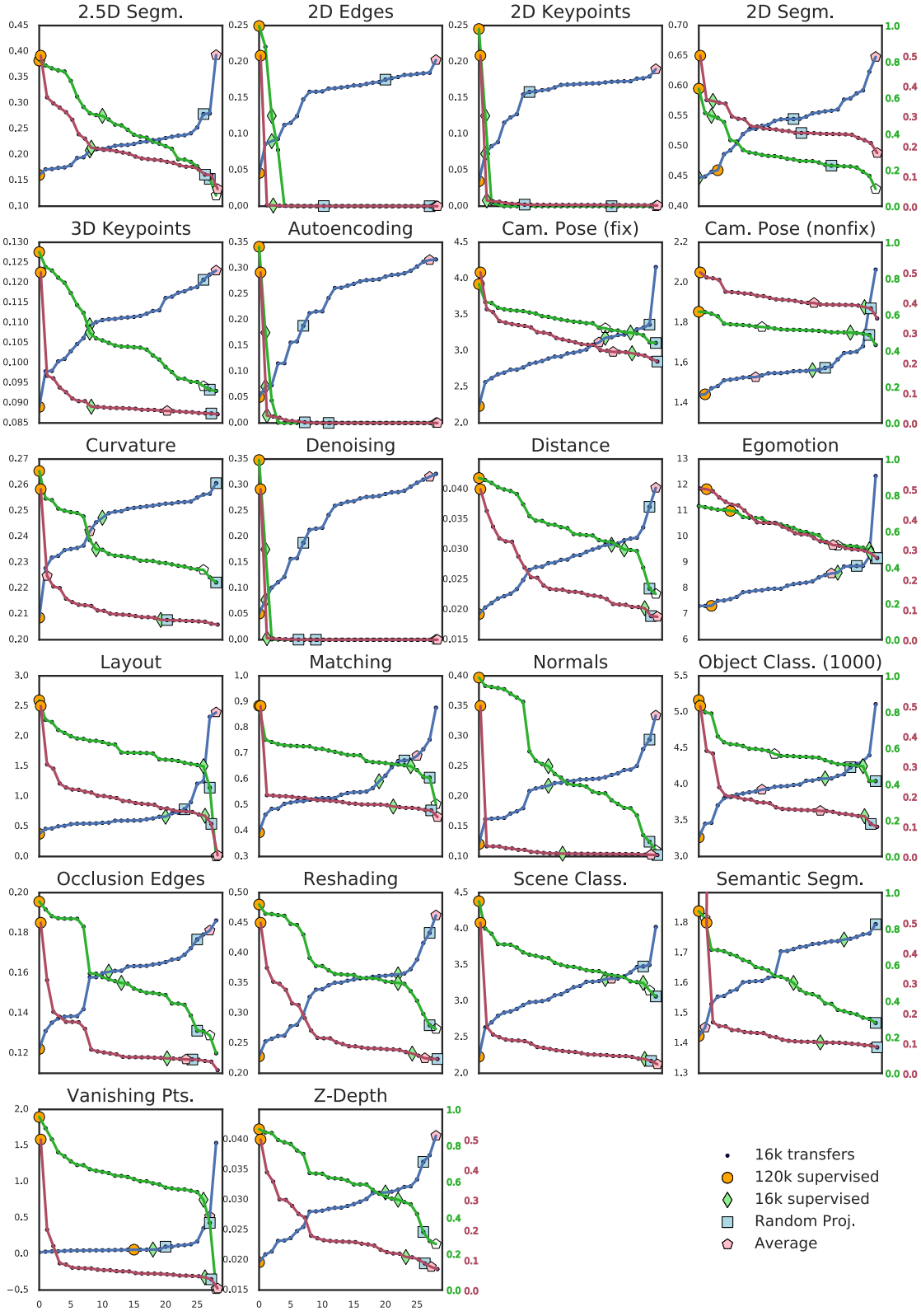
Surface Normal Estimation Surface normal estimation is thought to be crucial for spatial cognition. For example, objects can only be placed on surfaces with upwards-facing normals. Even for locomotion, a point with horizontal-facing normals indicates that it cannot be easily traversed. Surface normals are computed directly from the 3D mesh.

Vanishing Point Estimation An consequence of perspective, vanishing points offer useful information about the scene geometry [26, 17] and are well studied. Vanishing points prove particularly useful in a Manhattan world [7, 49, 5] where there are three dominant vanishing points corresponding to an X, Y, and Z axis. This assumption is usually met in urban environments. For each model we analytically find these three vanishing points and include them as labels.

Semantic Learning through Knowledge distillation While our dataset does not include semantic annotations, semantic understanding comprises a large and important component of modern computer vision. Therefore, we add pseudo-semantic annotations through knowledge distillation [13]. We distill knowledge from state-of-the-art models [10, 20] trained on ImageNet [33] and MS-COCO [22] by using them to annotate our dataset and then supervising models with those annotations. See section 13 for the details of knowledge distillation process.

1. **Classification, Semantic (1000-classes)** Semantic object recognition is a fundamental component of visual perception. Children learn at an early age to classify objects after seeing just a handful of examples [**?**]. For semantic classification, we distill knowledge form a pretrained ResNet-152 [10] (trained on ImageNet) by supervising our model with the ResNet activations.
2. **Classification Semantic (100-classes)** Many of the classes in ImageNet never appear in our dataset (e.g. animals, sports). We therefore manually select classes which appear in our dataset – and this happened to be 100 classes. For 100-way classification, we distill knowledge from these 100 activations only.
3. **Segmentation, Semantic** Models and agents need to know more than just what they are looking at – they also need to be able to locate the object. Therefore, we include knowledge distillation from the semantic segmentation model in [20], trained on MS COCO. [34]

Sorted **Quality**, **Gain**, and **Loss** of 1st order transfers for each task



References

[1] E. H. Adelson and A. P. Pentland. The perception of shading and reflectance. *Perception as Baysian Inference*, pages pp. 409–423, 1996. 6

[2] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 37–45, 2015. 4

[3] J. T. Barron and J. Malik. Shape, illumination, and reflectance from shading. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(8):1670–1687, Aug 2015. 6

[4] H. Bay, T. Tuytelaars, and L. Van Gool. *SURF: Speeded Up Robust Features*, pages 404–417. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006. 5

[5] J. C. Bazin, Y. Seo, C. Demonceaux, P. Vasseur, K. Ikeuchi, I. Kweon, and M. Pollefeys. Globally optimal line clustering and vanishing point estimation in manhattan world. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 638–645, June 2012. 6

[6] A. Y.-S. Chia, S. Zhuo, R. K. Gupta, Y.-W. Tai, S.-Y. Cho, P. Tan, and S. Lin. Semantic colorization with internet images. *ACM Trans. Graph.*, 30(6):156:1–156:8, Dec. 2011. 5

[7] J. M. Coughlan and A. L. Yuille. The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 845–851. MIT Press, 2001. 6

[8] D. DeTone, T. Malisiewicz, and A. Rabinovich. Deep image homography estimation. *CoRR*, abs/1606.03798, 2016. 6

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012. 6

[10] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 6

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[12] R. Held and A. Hein. Movement-produced stimulation in the development of visually guided behavior. *J Comp Physiol Psychol*, pages 872–876. 6

[13] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 5, 6

[14] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 5

[15] H. Izadinia, Q. Shan, and S. M. Seitz. IM2CAD. *CoRR*, abs/1608.05137, 2016. 6

[16] J. Knopp, M. Prasad, G. Willems, R. Timofte, and L. Van Gool. *Hough Transform and 3DSURF for Robust ThreeDimensional Classification*, pages 589–602. Springer Berlin Heidelberg, Berlin, Heidelberg, 2010. 5

[17] H. Kong, J. Y. Audibert, and J. Ponce. Vanishing point detection for road detection. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 96–103, June 2009. 6

[18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012. 4

[19] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *3D Vision (3DV), 2016 Fourth International Conference on*, pages 239–248. IEEE, 2016. 3

[20] Y. Li, H. Qi, J. Dai, X. Ji, and Y. Wei. Fully convolutional instance-aware semantic segmentation. *arXiv preprint arXiv:1611.07709*, 2016. 6

[21] 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*, abs/1405.0312, 2014. 5

[22] 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*, abs/1405.0312, 2014. 6

[23] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999. 6

[24] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004. 5

[25] A. Mian, M. Bennamoun, and R. Owens. On the repeatability and quality of keypoints for local feature-based 3d object retrieval from cluttered scenes. *International Journal of Computer Vision*, 89(2):348–361, Sep 2010. 5

[26] O. Miksik. Rapid vanishing point estimation for general road detection. In *2012 IEEE International Conference on Robotics and Automation*, pages 4844–4849, May 2012. 6

[27] N. D. Molton, A. J. Davison, and I. D. Reid. Locally planar patch features for real-time structure from motion. In *Proc. British Machine Vision Conference*. BMVC, Sept. 2004. (To appear). 6

[28] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages I–652–I–659 Vol.1, June 2004. 6

[29] M. Noroozi and P. Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 4, 5

[30] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. 2016. 5

[31] N. Rader, M. Bausano, and J. E. Richards. On the nature of the visual-cliff-avoidance response in human infants. *Child Development*, 51(1):61–68, 1980. 6

[32] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV

'11, pages 2564–2571, Washington, DC, USA, 2011. IEEE Computer Society. 5

[33] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 4, 6

[34] A. Sax*, W. B. Shen*, A. R. Zamir*, L. J. Guibas, J. Malik, and S. Savarese. Taskonomy: Disentangling task transfer learning. *CVPR*, 2017. submitted. 6

[35] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks, 2002. 6

[36] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 519–528, June 2006. 6

[37] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, Aug. 2000. 6

[38] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. *Indoor Segmentation and Support Inference from RGBD Images*, pages 746–760. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. 6

[39] B. Steder, R. B. Rusu, K. Konolige, and W. Burgard. Narf: 3d range image features for object recognition. 5

[40] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pages 1096–1103, New York, NY, USA, 2008. ACM. 5

[41] L. Wang, U. Neumann, and S. You. Wide-baseline image matching using line signatures. In *ICCV*, pages 1311–1318. IEEE Computer Society, 2009. 6

[42] X. Wang and A. Gupta. Unsupervised learning of visual representations using videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2015. 4

[43] M. Wertheimer. Laws of organization in perceptual forms. *Psycologische Forschung*, 4:301–350, 1923. 6

[44] B. Yao, G. Bradski, and L. Fei-Fei. A codebook-free and annotation-free approach for fine-grained image categorization. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3466–3473, June 2012. 6

[45] F. Yu, Y. Zhang, S. Song, A. Seff, and J. Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 6

[46] X. Yue, I. S. Pourladian, R. B. H. Tootell, and L. G. Ungerleider. Curvature-processing network in macaque visual cortex. *Proceedings of the National Academy of Sciences*, 111(33):E3467–E3475, 2014. 5

[47] A. Zaharescu, E. Boyer, K. Varanasi, and R. Horaud. Surface feature detection and description with applications to mesh matching. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 373–380, June 2009. 5

[48] A. R. Zamir, T. Wekel, P. Agrawal, C. Wei, J. Malik, and S. Savarese. Generic 3d representation via pose estimation and matching. In *European Conference on Computer Vision*, pages 535–553. Springer, 2016. 4

[49] L. Zhang, H. Lu, X. Hu, and R. Koch. Vanishing point estimation and line classification in a manhattan world with a unifying camera model. *International Journal of Computer Vision*, 117(2):111–130, Apr 2016. 4, 5, 6

[50] Y. Zhong. Intrinsic shape signatures: A shape descriptor for 3d object recognition. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pages 689–696, Sept 2009. 5

[51] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 487–495. Curran Associates, Inc., 2014. 4, 5

[52] Z. Zhu, T. Oskiper, S. Samarasekera, R. Kumar, and H. S. Sawhney. Ten-fold improvement in visual odometry using landmark matching. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Oct 2007. 6