1 Room Layout Topology Definition

Fig 1 shows the eleven possible room layout topologies in a 2D image. This figure is taken from LSUN challenge official document. Note that it starts from index-0, but in the paper it starts from index-1 when referred.

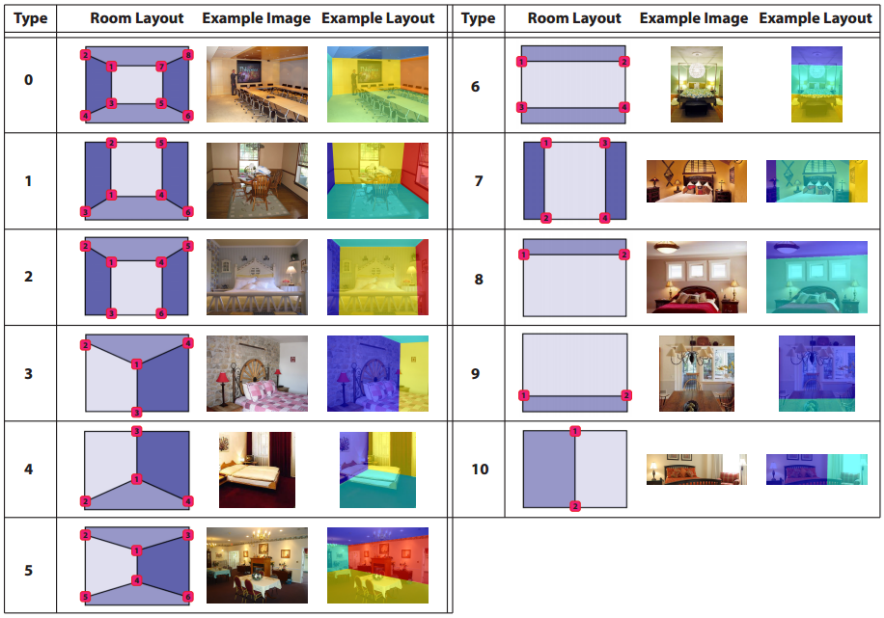


Figure 1: Eleven different room layout topologies.

2 Network Design

A building block of ResNet-101 is called a *bottleneck* (depicted as B later). A *bottleneck* is consisted of two branches. The first branch is the so-called *identity mapping* that directly feed forward the data blobs without change. The second branch is made up with three convolutional layers, whose spatial sizes are 1×1, 3×3 and 1×1 respectively. These two branches are summed at the end of the *bottleneck*. There are 5 *variant bottleneck* (depicted as V later) in ResNet-101. In a *variant bottleneck*, the *identity mapping* is replaced with a convolutional layer. ResNet-101's structure can be demonstrated as **conv1-V-2B-V-3B-V-22B-V-2B-fc**. In this network, *conv1* and each V down-size the feature map by a factor of 2. In order to modify this network into a FCN version, we do net surgery to: (1) the last two *variant bottlenecks*; (2) all the *bottlenecks* that follows last two *variant bottlenecks*; (3) the fc layer. The modified network can be demonstrated as **conv1-V-2B-V-3B-VS-22BH-VS-2BH-convH**. The meaning of VS, BH and convH will be explained later. The network design is illustrated by Fig 2.

As mentioned above, in original ResNet-101 there are five modules that would shrink the feature map, totally by a factor of 32. In order to compensate for resolution loss, feature map shrinking is eliminated for two out of these five modules, so that down-sampling factor is reduced to 8. More specifically, these two modules are the last two *variant bottlenecks*. As demonstrated by Fig 2, both branches in a V has a convolutional layer with a stride of 2. By setting the strides of them to 1, a V no longer down-samples the feature map and we name it as VS. However, this change violates the 3×3 convolutional layer's relationship with the feature map, since its parameters are pre-trained with original ResNet-101 on a image classification task. In order to reuse those pre-trained parameters, the *hole* mechanism of (2) is used. Let's take the extreme case for example. If the 3×3 layer (w_{ij}) formerly operates on a 3×3 feature map (f_{ij}), now it has to operate on a 5×5 feature map (F_{ij}). Former convolution operation actually calculates $w_{11}f_{11} + w_{12}f_{12} + w_{13}f_{13} + w_{21}f_{21} + w_{22}f_{22} + w_{23}f_{23} + w_{31}f_{31} + w_{32}f_{32} + w_{33}f_{33}$. In order to maintain this relationship on F_{ij}, a convolutional layer with *hole* calculates $w_{11}F_{11} + w_{12}F_{13} + w_{13}F_{15} + w_{21}F_{31} + w_{22}F_{33} + w_{23}F_{35} + w_{31}F_{51} +$

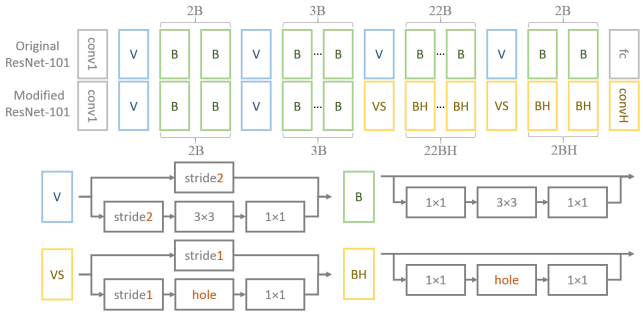


Figure 2: Network design for semantic segmentation pre-training.

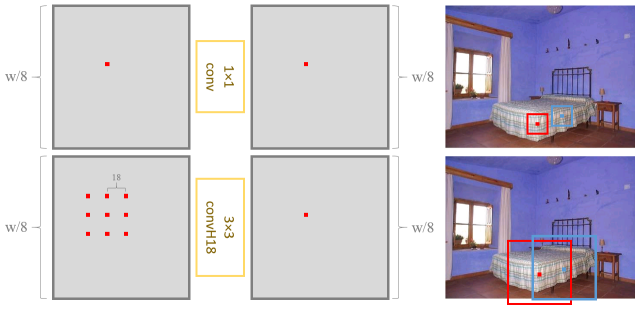


Figure 3: A 3×3 convolutional layer with 18 holes enlarges receptive field from 8×8 to 320×320. Patches in the small receptive field are not discriminative for edge labels.

$w_{32}F_{53} + w_{33}F_{55}$. Samples like F_{12} or F_{44} are skipped just like there are holes in these locations. It is also known as *dilated convolution* in (7) and now already a common practice for FCN design. Similarly, B is modified into BH by replacing the original 3×3 convolutional layer with a new 3×3 convolutional layer with *hole*. Note only bottlenecks following the last two variant bottlenecks need this net surgery.

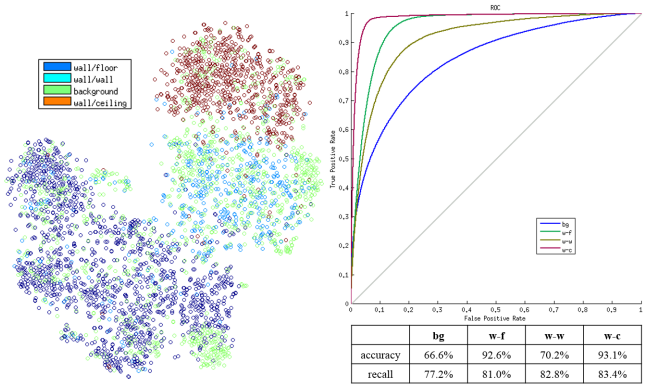
The fc layer is replaced with a spatially 3×3 convolutional layer with 18 holes (convH18), which can enlarge receptive field (Fig 3). As the figure's first row shows, if we use a spatially 1×1 convolutional layer, every output location is related to a single location in the input. Since the compensated feature map has a $w/8 \times w/8$ resolution, the whole network's receptive field is 8×8. Yet as the figure's second row shows, with a convH18 every output location is calculated from 9 elements in the input. More importantly, these 9 elements are separated by 18 holes so that the whole network's receptive field is 320×320 (formula as $((18 \times 2) + 3) \times 8 + 8 = 320$). Edge labeling in nature requires large receptive field. In Fig 3's last column, two typical pixels are used for demonstration. A background pixel and its corresponding receptive field are colored in red. A wall-floor edge pixel and its corresponding receptive field are colored in blue. In the upper picture, both the red box and the blue box only cover pixels within the *bed*. Color or texture features are basically the same in these two patches. It is difficult for an FCN to assign different labels to them. Even human vision system can hardly tell the difference between these two patches. However, in the bottom picture, enlarged receptive field provides more discriminative context information for those two target pixels, for example how many wall pixels there are. This 320×320 receptive field is maintained until stage three, since the transfer weights are reshaped into a $1 \times 1 \times 37 \times 4$ layer which won't change the receptive field.

3 Quantitative Semantic Segmentation Results

Table 1 shows the quantitative improvements of our network over baselines.

Table 1: Class-wise accuracy for 37-class semantic segmentation task on SUNRGBD test set.

method	wall	floor	cabi	bed	chair	sofa	table	door	wndw	bkshf	pic	blinds	cntr
(5)	43.2	78.6	26.2	42.5	33.2	40.6	34.3	33.2	43.6	23.1	57.2	31.8	**42.3**
(1)	86.8	92.0	52.4	68.4	76.0	54.3	59.3	37.4	53.8	29.2	49.7	32.5	31.2
ours	**89.9**	**94.0**	**62.4**	**78.5**	**83.1**	**66.5**	**67.5**	**48.8**	**63.2**	**47.3**	**59.6**	**41.1**	39.2
method	desk	shelf	curtn	dresr	pilw	mirror	flrmat	cloth	ceil	books	fridge	tv	paper
(5)	12.1	18.4	**59.1**	31.4	**49.5**	24.8	**5.6**	27.0	**84.5**	35.7	24.2	36.5	**26.8**
(1)	17.8	5.3	53.2	28.8	36.5	29.6	0.0	14.4	67.7	32.4	10.2	18.3	19.2
ours	**27.6**	**20.3**	55.6	**55.2**	49.4	**47.8**	0.0	**37.2**	78.4	**43.3**	**46.8**	**67.6**	24.1
method	towel	shwcn	box	whtbd	persn	ntstd	toilet	sink	lamp	btub	bag	mIU	mAC
(5)	19.2	**9.0**	11.7	51.4	35.7	**25.0**	64.1	53.0	**44.2**	47.0	18.6	—	36.3
(1)	11.5	0.0	8.9	38.7	4.9	22.6	55.6	52.7	27.9	29.9	8.1	26.3	35.6
ours	**32.6**	0.6	**35.6**	**59.4**	**60.8**	23.6	**81.7**	**67.7**	40.9	**53.6**	**23.3**	**40.0**	**50.6**



	bg	w-f	w-w	w-c
accuracy	66.6%	92.6%	70.2%	93.1%
recall	77.2%	81.0%	82.8%	83.4%

Figure 4: Left figure: unsupervised structure visualization of the semantic feature space. Right figure and table: receiver operating characteristic curve, class-wise accuracy, and class-wise recall on LSUN validation set.

4 Semantic Transfer

In ST's stage two we use dataset LSUN, which is the largest dataset with human annotated room layout. Dataset SUNRGBD is even larger but its room layout annotations are generated from depth images. LSUN is consisted of 4000 training images and 394 validation images. Original room layout ground truths of LSUN are in the format of P_i and E_i (see notations in the main paper). We convert them into pixel-wise edge label maps with $M = C(P_i, E_i)$. With the semantic segmentation network in stage one, a 37-channel semantic feature is also extracted for every pixel. We collect billions of samples from LSUN's training set, with every sample corresponding to a single pixel. This dataset is called LT. With t-sne (6), we visualize the structure of LT in an unsupervised way, in order to show that it's possible to bridge semantic features and edge labels. Further we learn a 37×4 fc layer to classify edge labels with semantic features. The training is done with Caffe (3) and stochastic gradient descent method (SGD). On LSUN's validation set we collect samples in the same way, and the dataset is called LV. on LV, we test the learnt fc layer. Similar to the observations in unsupervised analysis, classifying bg and w-w is more difficult than classifying w-c and w-f. Mean classification accuracy on LV is about 83%. Parameters of this fc layer will be used as initialization weights in ST's stage three.

5 Feature Quality and Comparison

Our pixel-wise edge labelling network produces highly robust features in all types of scenes, as visualized by Fig 5's left panel. All these samples are collected from LSUN test set, covering different types of scene clutter, layout configurations, and illumination conditions. We also provide feature quality comparisons against (4)'s failure cases, as demonstrated by Fig 5's right panel. These three scenes are very challenging because almost all edge pixels are occluded by clutters like sofas,

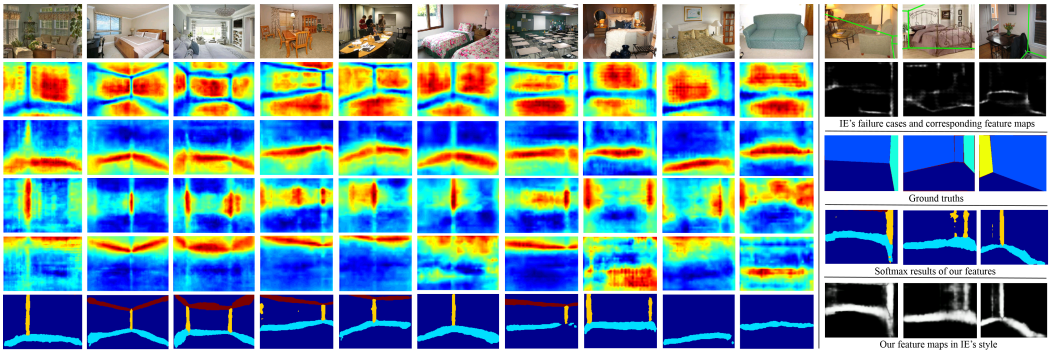


Figure 5: Left: feature quality visualization. From top to bottom: input, bg activation, wf activation, ww activation, wc activation, softmax results. (Note that every feature map is numerically normalized independently.) Right: feature quality comparison. (4)'s failure cases are clipped from original paper. For a fair comparison, feature maps produced by our network are processed into (4)'s style.

tables or beds. (4)'s network, which does not consider the relationship between room layout and scene clutter, fails to extract reliable edge features, yet our network handles them properly.

As shown by Fig 5, on these features it is possible to estimate room layouts directly with image processing techniques. We investigate the possibility of applying hough transformation to eroded softmax results and the qualitative results are not bad. However, using a series of image processing modules for inference is not a good choice, as many threshold parameters related to erosion or hough transformation have to be tuned. Instead, we propose to optimize a parameterized layout representation with those features. With similar consistency objective functions, we proposed two different implementations: naive optimization (NO) and physics inspired optimization (PIO).

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv:1511.00561*.

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016.

[3] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM international conference on Multimedia 2014*.

[4] A. Mallya and S. Lazebnik. Learning informative edge maps for indoor scene layout prediction. In *ICCV 2015*.

[5] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *CVPR 2015*.

[6] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research 2008*.

[7] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. *ICLR 2016*.