RoomNet: End-to-End Room Layout Estimation

Chen-Yu Lee Vijay Badrinarayanan Tomasz Malisiewicz Andrew Rabinovich
Magic Leap, Inc.
{clee,vbadrinarayanan,tmalisiewicz,arabinovich}@magicleap.com

Abstract

This paper focuses on the task of room layout estimation from a monocular RGB image. Prior works break the problem into two sub-tasks: semantic segmentation of floor, walls, ceiling to produce layout hypotheses, followed by an iterative optimization step to rank these hypotheses.

In contrast, we adopt a more direct formulation of this problem as one of estimating an ordered set of room layout keypoints. The room layout and the corresponding segmentation is completely specified given the locations of these ordered keypoints. We predict the locations of the room layout keypoints using RoomNet, an end-to-end trainable encoder-decoder network. On the challenging benchmark datasets Hedau and LSUN, we achieve state-of-the-art performance along with 200× to 600× speedup compared to the most recent work. Additionally, we present optional extensions to the RoomNet architecture such as including recurrent computations and memory units to refine the keypoint locations under the same parametric capacity.

1. Introduction

Room layout estimation from a monocular image, which aims to delineate a 2D boxy representation of an indoor scene, is an essential step for a wide variety of computer vision tasks, and has recently received great attention from several applications. These include indoor navigation [29], scene reconstruction/rendering [19], and augmented reality [46, 25, 10].

The field of room layout estimation has been primarily focused on using bottom-up image features such as local color, texture, and edge cues followed by vanishing point detection. A separate post-processing stage is used to clean up feature outliers and generate/rank a large set of room layout hypotheses with structured SVMs or conditional random fields (CRFs) [15, 11, 16, 36, 49]. In principle, the 3D reconstruction of the room layout can be obtained (up to scale) with knowledge of the 2D layout and the vanishing points. However, in practice, the accuracy of the final layout prediction often largely depends on the quality of the extracted low-level image features, which in itself is susceptible to local noise, scene clutter and occlusion.

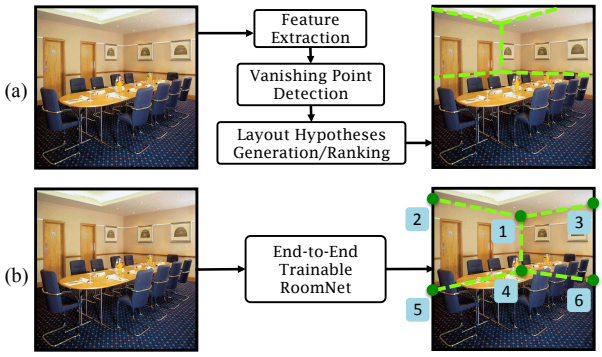

Figure 1. (a) Typical multi-step pipeline for room layout estimation. (b) Room layout estimation with RoomNet is direct and simple: run RoomNet, extract a set of room layout keypoints, and connect the keypoints in a specific order to obtain the layout.

Recently, with the rapid advances in deep convolutional neural networks (CNNs) for semantic segmentation [5, 27, 32, 2], researchers have been exploring the possibility of using such CNNs for room layout estimation. More specifically, Mallya *et al*. [28] first train a fully convolutional network (FCN) [27] model to produce "informative edge maps" that replace hand engineered low-level image feature extraction. The predicted edge maps are then used to sample vanishing lines for layout hypotheses generation and ranking. Dasgupta *et al*. [7] use the FCN to learn semantic surface labels such as left wall, front wall, right wall, ceiling, and ground. Then connected components and hole filling techniques are used to refine the raw per pixel prediction of the FCN, followed by the classic vanishing point/line sampling methods to produce room layouts. However, despite the improved results, these methods use CNNs to generate a new set of "low-level" features and fall short of exploiting the end-to-end learning ability of CNNs. In other words, the raw CNN predictions need to be post-processed by an expensive hypotheses testing stage to produce the final layout. This, for example, takes the pipeline of Dasgupta *et al*. [7] 30 seconds to process each frame.

In this work, we address the problem top-down by di-

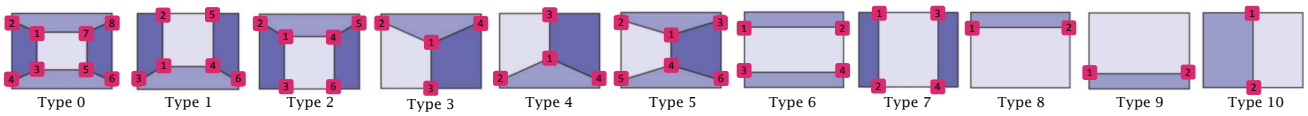

Figure 2. Definition of room layout types. The type is indexed from 0 to 10 as in [50]. The number on each keypoint defines the specific order of points saved in the ground truth. For a given room type, the ordering of keypoints specifies their connectivities.

rectly training CNNs to infer both the room layout corners (keypoints) and room type. Once the room type is inferred and the corresponding set of ordered keypoints are localized, we can connect them in a specific order to obtain the 2D spatial room layout. The proposed method, RoomNet, is direct and simple as illustrated in Figure 1: The network takes an input image of size 320×320, processes it through a convolutional encoder-decoder architecture, extracts a set of room layout keypoints, and then simply connects the obtained keypoints in a specific order to draw a room layout. The semantic segmentation of the layout surfaces is simply obtainable as a consequence of this connectivity.

Overall, we make several contributions in this paper: (1) reformulate the task of room layout estimation as a keypoint localization problem that can be directly addressed using CNNs, (2) a custom designed convolutional encoder-decoder network, RoomNet, for parametrically efficient and effective joint keypoint regression and room layout type classification, and (3) state-of-the-art performance on challenging benchmarks Hedau [15] and LSUN [50] along with $200\times$ to $600\times$ speedup compared to the most recent work.

2. RoomNet

2.1. Keypoint-based room layout representaiton

To design an effective room layout estimation system, it is important to choose a proper target output representation that is end-to-end trainable and can be inferred efficiently. Intuitively, one can assign geometric context classes (floor, walls, and ceiling) to each pixel, and then try to obtain room layout keypoints and boundaries based on the pixel-wised labels. However, it is non-trivial to derive layout keypoints and boundaries from the raw pixel output. In contrast, if we can design a model that directly outputs a set of ordered room layout keypoint locations, it is then trivial to obtain both keypoint-based and pixel-based room layout representations.

Another important property of using a keypoint-based representation is that it eliminates the ambiguity in the pixel-based representation. Researchers have shown that CNNs often have difficulty distinguishing between different surface identities. For instance, CNNs can be confused between the front wall class and the right wall class, and thereby output irregular or mixed pixel-wise labels within the same surface – this is well illustrated by Figure 5 and 6 from [7]. This phenomenon also largely undermines the overall room layout estimation performance.

Hence, we propose to use a keypoint-based room layout representation to train our model. Figure 2 shows a list of room types with their respective keypoint definition as defined by [50]. These 11 room layouts cover most of the possible situations under typical camera poses and common cuboid representations under "Manhattan world assumption" [6]. Once the trained model predicts correct keypoint locations with an associated room type, we can then simply connect these points in a specific order to produce boxy room layout representation.

2.2. Architecture of RoomNet

We design a CNN to delineate room layout structure using 2D keypoints. The input to the network is a single RGB image and the output of the network is a set of 2D keypoints in a specific order with an associated room type.

Keypoint estimation The base network architecture for keypoint estimation is inspired by the recent successes in the field of semantic segmentation [27, 32, 2]. Here we adopt the SegNet architecture proposed by Badrinarayanan *et al*. [1, 2] with modifications. Initially designed for segmentation, the SegNet framework consists of encoder and decoder sub-networks – the encoder of the SegNet maps an input image to lower resolution feature maps, and then the role of the decoder is to upsample the low resolution encoded feature maps to full input resolution for pixel-wise classification. In particular, the decoder uses pooling indices computed in the max-pooling step of the corresponding encoder to perform non-linear upsampling. This eliminates the need for learning to upsample. The upsampled maps are sparse and are convolved with trainable filters to produce dense feature map. This architecture has proven to provide good performance with competitive inference time and efficient memory usage as compared to other recent semantic segmentation architectures.

The base architecture of RoomNet adopts essentially the same convolutional encoder-decoder network as in SegNet. It takes an image of an indoor scene and directly outputs a set of 2D room layout keypoints to recover the room layout structure. Each keypoint ground truth is represented by a 2D Gaussian heatmap centered at the true keypoint location as one of the channels in the output layer.[1] The encoder-decoder architecture processes the information flow through bottleneck layers, enforcing it to implicitly model the rela-

[1]We color-code and visualize multiple keypoint heatmaps in a single 2D image in Figure 3, Figure 5 and the rest of the paper.

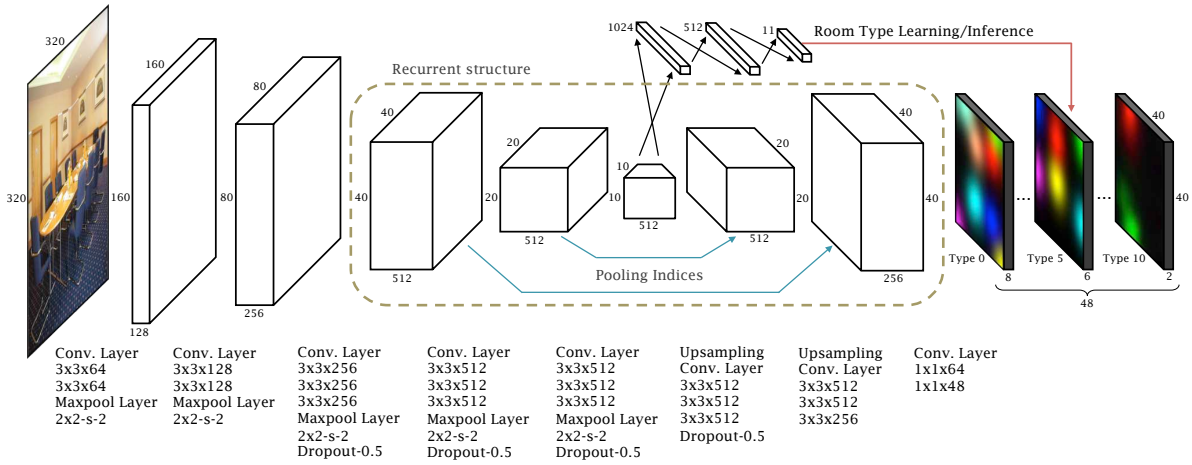

Figure 3. An illustration of the RoomNet base architecture. A decoder upsamples its input using the transferred pooling indices from its encoder to produce sparse feature maps followed by a several convolutional layers with trainable filter banks to densify the feature responses. The final decoder output keypoint heatmaps are fed to a regressor with Euclidean losses. A side head with 3 fully-connected layers is attached to the bottleneck layer and used to train/predict the room type class label, which is then used to select the associated set of keypoint heatmaps. The full model of RoomNet with recurrent encoder-decoder (center dashed line block) further performs keypoint refinement as shown in Figure 4 (b) and 5.

tionship among the keypoints that encode the 2D structure of the room layout.

The decoder of the RoomNet upsamples the feature maps from the bottleneck layer with spatial dimension 10 $\times$ 10 to 40 $\times$ 40 instead of the full resolution 320 $\times$ 320 as shown in Figure 3. This is because we empirically found that using the proposed 2D keypoint-based representation can already model the room layout effectively at 40 $\times$ 40 scale (results are similar as compared to training decoder sub-network at full resolution). Using this "trimmed" decoder sub-network also significantly reduces the memory/time cost during both training and testing due to the high computation cost of convolution at higher resolutions.

Extending to multiple room types The aforementioned keypoint estimation framework serves as a basic room layout estimation system for one particular room type. To generalize this approach for multiple room types, one possible solution is to train one network per class as in the Single Image 3D Interpreter Network of Wu *et al*. [45]. However, in order to maximize efficiency, we design the RoomNet to be fast from the ground up. Encouraged by the recent object detection works YOLO [37] and SSD [26] that utilize a single neural network to predict bounding boxes and class probabilities directly from full images in one evaluation, our proposed RoomNet similarly predicts room layout keypoints and the associated room type with respect to the input image in one forward pass. To achieve this goal, we increase the number of channels in the output layer to match the total number of keypoints for all 11 room types (total 48 keypoints for 11 room types derived from Figure 2), and we also add a side head with fully connected layers to the bottleneck layer (the layer where usually used for image classification) for room type prediction as shown in Figure 3.

We denote a training example as $(\boldsymbol{\mathcal{I}}, \boldsymbol{y}, t)$, where $\boldsymbol{y}$ stands for the ground truth coordinates of the k keypoints with room type t for the input image $\boldsymbol{\mathcal{I}}$. At training stage, we use the Euclidean loss as the cost function for layout keypoint heatmap regression and use the cross-entropy loss for the room type prediction. Given the keypoint heatmap regressor φ (output from the decoder sub-network), and the room type classifier ψ (output from the fully-connected side head layer), we can then optimize the following loss function:

$$\sum_{k} \mathbb{1}_{k,t}^{\text{keypoint}} \|G_k(\boldsymbol{y}) - \varphi_k(\boldsymbol{\mathcal{I}})\|^2 - \lambda \sum_{c} \mathbb{1}_{c,t}^{\text{room}} \log(\psi_c(\boldsymbol{\mathcal{I}})) \quad (1)$$

where $\mathbb{1}_{k,t}^{\text{keypoint}}$ denotes if keypoint k appears in ground truth room type t, $\mathbb{1}_{c,t}^{\text{room}}$ denotes if room type index c equals to the ground truth room type t, G is a Gaussian centered at $\boldsymbol{y}$ and the weight term λ is set to 5 by cross validation. The first term in the loss function compares the predicted heatmaps to ground-truth heatmaps synthesized for each keypoint separately. The ground truth for each keypoint heatmap is a 2D Gaussian centered on the true keypoint location with standard deviation of 5 pixels as in the common practice in recent keypoint regression works [43, 35, 4, 45]. The second term in the loss function encourages the side head fully-connected layers to produce a high confidence value with respect to the correct room type class label.

Note that one forward pass of the proposed architecture will produce keypoint heatmaps for all room types. However, the loss function only penalizes Euclidean regression error if the keypoint k is present for the ground truth room

type t in the current input image $\mathcal{I}$, effectively using the predicted room type indices to select the corresponding set of keypoint heatmaps to update the regressor. The same strategy applies at the test stage *i.e.* the predicted room type is used to select the corresponding set of keypoint heatmaps in the final output.

RoomNet extension for keypoint refinement Recurrent neural networks (RNNs) and its variant Long Short-Term Memory (LSTM) [17] have proven to be extremely effective models when dealing with sequential data. Since then, researchers have been exploring the use of recurrent structures for static input format as well, such as recurrent convolutional layers [24] and convLSTM layers [48].

Recently, more sophisticated iterative/recurrent architectures have been proposed for 2D static input, such as FCN with CRF-RNN [52], iterative error feedback networks [4], recurrent CNNs [3], stacked encoder-decoder [31], and recurrent encoder-decoder networks [34, 22]. These evidence show that adopting the "time series" concept when modeling a static input can also significantly improve the ability of the network to integrate contextual information and to reduce prediction error.

Motivated by the aforementioned successes, we extend our base RoomNet architecture by making the central encoder-decoder component (see center dashed line block in Figure 3) recurrent. Specifically, we propose a memory augmented recurrent encoder-decoder (MRED) structure (see Figure 4 (b)) whose goal is to mimic the behavior of a typical recurrent neural network (Figure 4 (a)) in order to refine the predicted keypoint heatmaps over "time" – the artificial time steps created by the recurrent structure.

Each layer in this MRED structure shares the same weight matrices through different time steps that convolve (denoted as $*$ symbol) with the incoming feature maps from the previous prediction $\mathbf{h}_l(t-1)$ at time step $t-1$ in the same layer l and the current input $\mathbf{h}_{l-1}(t)$ at time step t in the previous layer $l-1$, generating output at time step t as:

$$\mathbf{h}_l(t) = \begin{cases} \sigma(\mathbf{w}_l^{\text{current}} * \mathbf{h}_{l-1}(t) + \mathbf{b}_l) & , t = 0 \\ \sigma(\mathbf{w}_l^{\text{current}} * \mathbf{h}_{l-1}(t) + \mathbf{w}_l^{\text{previous}} * \mathbf{h}_l(t-1) + \mathbf{b}_l) & , t > 0 \end{cases} \tag{2}$$

where $\mathbf{w}_l^{\text{current}}$ and $\mathbf{w}_l^{\text{previous}}$ are the input and feed-forward weights for layer l. $\mathbf{b}_l$ is the bias for layer l. σ is the ReLU activation function [30].

Figure 4 (b) demonstrates the overall process of the information flow during forward- and backward- propagations through depth and time within the recurrent encoder-decoder structure. The advantages of using the proposed MRED architecture are: (1) exploiting the contextual and structural knowledge among keypoints iteratively through hidden/memory units (that have not been explored in recurrent convolutional encoder-decoder structure) and (2) weight sharing of the convolutional layers in the recurrent encoder-decoder, resulting in a much deeper network with a

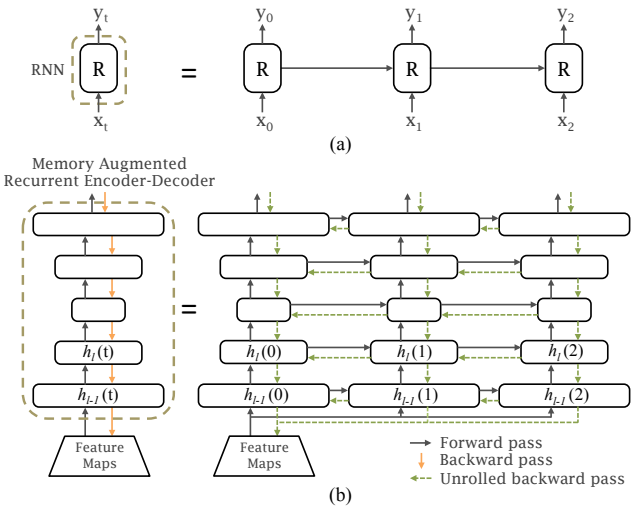

Figure 4. Illustration of unrolled (3 iterations) version of (a) a RNN and (b) the proposed memory augmented recurrent encoder-decoder architecture that mimics the behavior of a RNN but which is designed for a static input. Both structures have hidden units to store previous activations that help the inference at the current time step.

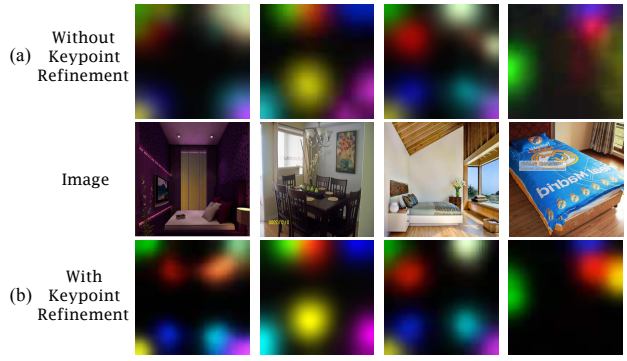

Figure 5. Room layout keypoint estimation from a single image (a) without refinement and (b) with refinement. Keypoint heatmaps from multiple channels are color-coded and shown in a single 2D image for visualization purposes. The keypoint refinement step produces more concentrated and cleaner heatmaps and removes some false positives.

fixed number of parameters. After refinement, the heatmaps of keypoints are much cleaner as shown in Figure 5. It is also interesting to observe the mistakes made early on and corrected later by the network (see third and fourth columns in Figure 5). We analyze the performance with and without the keypoint refinement step in Section 3.4, and we also evaluate different encoder-decoder variants in Section 4.

Deep supervision through time When applying stacked, iterative, or recurrent convolutional structures, each layer in the network receives gradients across more layers or/and time steps, resulting in models that are much harder to train. For instance, the iterative error feedback network [4] re-

quires multi-stage training and the stacked encoder-decoder structure in [31] uses intermediate supervision at the end of each encoder-decoder even when batch normalization [18] is used. Following the practices in [23, 42], we extend the idea by injecting supervision at the end of each time step. The same loss function L is applied to all the time steps as demonstrated in Figure 6. Section 3.4 and Table 5 provide details of the analysis and effect of the deep supervision through time.

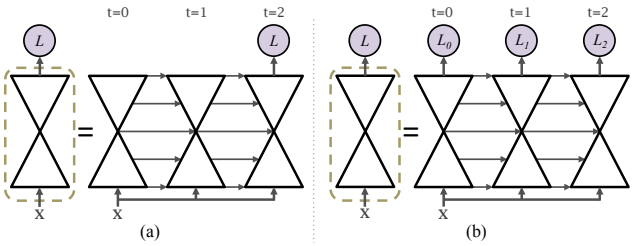

Figure 6. Illustration of the proposed memory augmented recurrent encoder-decoder architecture (a) without deep supervision through time and (b) with deep supervision through time.

3. Experiments

3.1. Datasets

We evaluate the proposed RoomNet framework on two challenging benchmark datasets: Hedau [15] dataset and Large-scale Scene Understanding Challenge (LSUN) room layout dataset [50]. The Hedau dataset contains 209 training, 53 validation, and 105 test images that are collected from the web and from LabelMe [39]. The LSUN dataset consists of 4000 training, 394 validation, and 1000 test images that are sampled from SUN database [47]. We follow the same experimental setup as Dasgupta *et al*. [7]. We rescale all input images to 320 $\times$ 320 pixels and train our network from scratch on the LSUN training set only. All experimental results are computed using the LSUN room layout challenge toolkit [50] on the original image scales.

3.2. Implementation details

The input to the network is an RGB image of resolution 320 $\times$ 320 and the output is the room layout keypoint heatmaps of resolution 40 $\times$ 40 with an associated room type class label. We apply the backpropagation through time (BPTT) algorithm to train the models with batch size 20 SGD, 0.5 dropout rate, 0.9 momentum, and 0.0005 weight decay. Initial learning rate is 0.00001 and decreased by a factor of 5 twice at epoch 150 and 200, respectively. All variants use the same scheme with 225 total epochs. The encoder and decoder weights are all initialized using the technique described in He *et al*. [13]. Batch normalization [18] and ReLU [30] activation function are also used after each convolutional layer to improve the training process. We apply horizontal flipping of input images during training as the only data augmentation. The system is implemented in the open source deep learning framework Caffe [20].

In addition, a ground truth keypoint heatmap has zero value (background) for most of its area and only a small portion of it corresponds to the Gaussian distribution (foreground associated with actual keypoint location). The output of the network therefore tends to converges to zero due to the imbalance between foreground and background distributions. For this reason, it is crucial to weight the gradients based on the ratio between foreground and background area for each keypoint heatmap. In our experiment, we degrade the gradients of background pixels by multiplying them with a factor of 0.2 and found this makes training significantly more stable.

Training from scratch takes about 40 hours on 4 NVIDIA Titan X GPUs. One forward inference of the full model (RoomNet recurrent 3-iter) takes 83 ms on a single GPU. For generating final test predictions we run both the original input and a flipped version of the image through the network and average the heatmaps together (accounting for a 0.12% average improvement on keypoint error and a 0.15% average improvement on pixel error) as in [31]. The keypoint location is chosen to be the max activating location of the corresponding heatmap.

3.3. Results

Two standard room layout estimation evaluation metrices are: (1) pixel error: pixelwise error between the predicted surface labels and ground truth labels, and (2) keypoint error: average Euclidean distance between the predicted keypoint and annotated keypoint locations, normalized by the image diagonal length.

Accuracy We summarize the performance on both datasets in Table 1 and 2. The previous best method is the two-step framework (per pixel CNN-based segmentation with a separate hypotheses ranking approach) Dasgupta *et al*. [7]. The proposed RoomNet significantly improves upon the previous results on both keypoint error and pixel error, achieving state-of-the-art performance [2].

To decouple the performance gains due to external data, we also prepare results of fine-tuning the RoomNet from a SUN [41] pre-trained model (on semantic segmentation task) and this achieves 6.09% keypoint error and 9.04% pixel error as compared of method in [38][3] with 7.95% keypoint error and 9.31% pixel error on LSUN dataset.

Runtime and complexity Efficiency evaluation on the input image size of 320 $\times$ 320 is shown in Table 3. Our full model (RoomNet recurrent 3 iteration) achieves 200$\times$ speedup compares to the previous best method in [7], and

[2]The side head room type classifier obtained 81.5% accuracy on LSUN dataset.

[3]The multi-step method in [38] utilizes additional Hedau+ [28] training set and fine-tunes from NYUDv2 RGBD [12] pre-trained models.

Method	Pixel Error (%)
Hedau *et al*. (2009) [15]	21.20
Del Pero *et al*. (2012) [8]	16.30
Gupta *et al*. (2010) [11]	16.20
Zhao *et al*. (2013) [51]	14.50
Ramalingam *et al*. (2013) [36]	13.34
Mallya *et al*. (2015) [28]	12.83
Schwing *et al*. (2012) [40]	12.8
Del Pero *et al*. (2013) [9]	12.7
Dasgupta *et al*. (2016) [7]	9.73
RoomNet recurrent 3-iter (ours)	**8.36**

Table 1. Performance on Hedau dataset [15]. We outperform the previous best result in [7] using the proposed end-to-end trainable RoomNet.

Method	Keypoint Error (%)	Pixel Error (%)
Hedau *et al*. (2009) [15]	15.48	24.23
Mallya *et al*. (2015) [28]	11.02	16.71
Dasgupta *et al*. (2016) [7]	8.20	10.63
RoomNet recurrent 3-iter (ours)	**6.30**	**9.86**

Table 2. Performance on LSUN dataset [50]. We outperform the previous best result in [7] on both keypoint and pixel errors using the proposed end-to-end trainable RoomNet.

Method	FPS
Del Pero *et al*. (2013) [9]	0.001
Dasgupta *et al*. (2016) [7]	0.03
RoomNet recurrent 3-iter	5.96
RoomNet recurrent 2-iter	8.89
RoomNet basic	19.26

Table 3. Runtime evaluation on an input size of 320×320. The proposed RoomNet full model (3-iter) achieves 200× speedup and the basic RoomNet model achieves 600× speedup than the previous best method in [7].

the base RoomNet without recurrent structure (RoomNet basic) achieves 600× speedup. Note that the timing is for two forward passes as described earlier. Using either one of the proposed architecture can provide significant inference time reduction and an improved accuracy as shown in Table 4.

3.4. Analyzing RoomNet

In this section, we empirically investigate the effect of each component in the proposed architecture with the LSUN dataset as our running example.

Recurrent vs direct prediction Table 4 shows the effectiveness of extending the RoomNet-basic architecture to a memory augmented recurrent encoder-decoder networks. We observed that more iterations led to lower error rates

Model	Keypoint Error (%)	Pixel Error (%)
RoomNet basic	6.95	10.46
RoomNet recurrent 2-iter	6.65	9.97
RoomNet recurrent 3-iter	**6.30**	**9.86**

Table 4. The impact of keypoint refinement step (see Section 2.2) using the proposed memory augmented recurrent encoder-decoder architecture on LSUN dataset [50].

Model	Keypoint Error (%)	Pixel Error (%)
RoomNet recurrent 2-iter		
- w/o deep supervision through time	6.93	10.44
- w/ deep supervision through time	6.65	9.97
RoomNet recurrent 3-iter		
- w/o deep supervision through time	6.95	10.47
- w/ deep supervision through time	**6.30**	**9.86**

Table 5. The impact of deep supervision through time on LSUN dataset [50] for RoomNets with 2 and 3 recurrent iterations.

on both keypoint error and pixel error: the RoomNet with recurrent structure that iteratively regresses to correct keypoint locations achieves 6.3% keypoint error and 9.86 pixel error as compared to the RoomNet without recurrent structure which achieves 6.95% keypoint error and 10.46 pixel error. No further significant performance improvement is observed after 3 iterations. Notice that the improvement essentially came from the same parametric capacity within the networks since the weights of convolutional layers are shared across iterations.

Importance of deep supervision through time When applying a recurrent structure with encoder-decoder architectures, each layer in the network receives gradients not only across depth but also through time steps between the input and the final objective function during training. It is therefore of interest to investigate the effect of adding auxiliary loss functions at different time steps. Table 5 demonstrates the impact of deep supervision through time using RoomNet with 2 and 3 recurrent iterations. We observed immediate reduction in both keypoint error and pixel error by adding auxiliary losses for both cases. This can be understood by the fact that the learning problem with deep supervision is much easier [23, 42] through different time steps. It is also interesting to point out that RoomNet 3-iter performs worse than RoomNet 2-iter when deep supervision through time is not applied. This is rectified when deep supervision through time is applied. Overall, we validate that with more iterations in the recurrent structure, there is a stronger need to apply deep supervision through time to successfully train the proposed architecture.

Qualitative results We show qualitative results of the proposed RoomNet in Figure 7. When the image is clean and

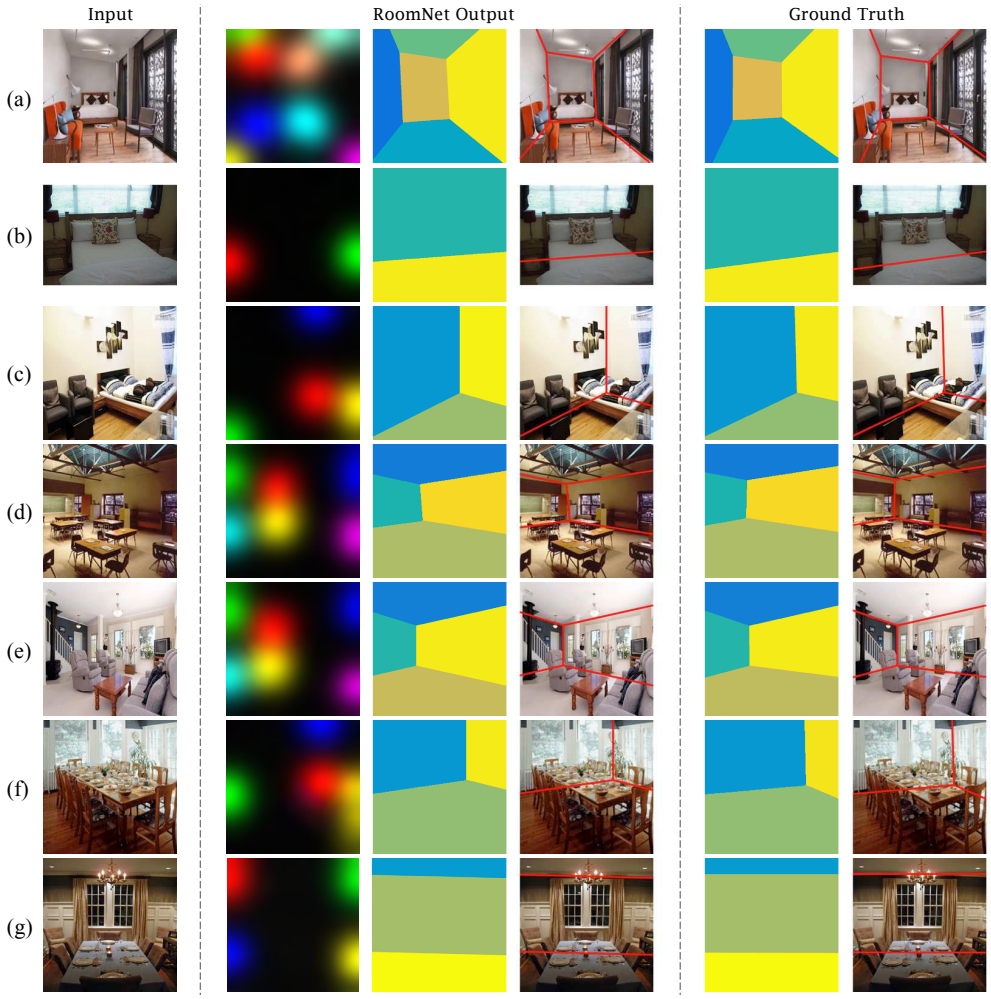

Figure 7. The RoomNet predictions and the corresponding ground truth on LSUN dataset. The proposed architecture takes a RGB input (first column) and produces room layout keypoint heatmaps (second column). The final keypoints are obtained by extracting the location with maximum response from the heatmaps. The third and fourth columns show a boxy room layout representation by simply connecting obtained keypoints in a specific order as in Figure 2. The fifth and sixth columns show the ground truth. Our algorithm is robust to keypoint occlusion by objects (ex: tables, chairs, beds).

the room layout boundaries/corners are not occluded, our algorithm can recover the boxy room layout representation with high accuracy. Our framework is also robust to keypoint occlusion by objects (ex: tables, chairs, beds), demonstrated in Figure 7 (b)(c)(d)(f). The major failure cases are when room layout boundaries are barely visible (Figure 8 (a)(c)) or when there are more than one plausible room layout explanations for a given image of a scene (Figure 8 (b)(d)).

4. Discussion

Alternative encoder-decoders We provide an evaluation of alternative encoder-decoder architectures for the room layout estimation task including: (a) a vanilla encoder-decoder (RoomNet basic), (b) stacked encoder-decoder, (c) stacked encoder-decoder with skip-connections; (d) encoder-decoder with feedback; (e) memory augmented recurrent encoder-decoder (RoomNet full); (f) memory augmented recurrent encoder-decoder with feedback. Figure 9 illustrates the 6 different network configurations that are evaluated here. We emphasize that our intention is not to put each encoder-decoder variant in competition, but to provide an illustrative comparison of the relative benefits of different configurations for the task being addressed here. Table 6 shows the performance of different variants on LSUN dataset.

The comparison of (a) and (b) variants indicates that stacking encoder-decoder networks can further improve the performance, as the network is enforced to learn the spatial structure of the room layout keypoints implicitly by placing constraints on multiple bottleneck layers.

However, adding skip connections [14, 31] as in (c) does

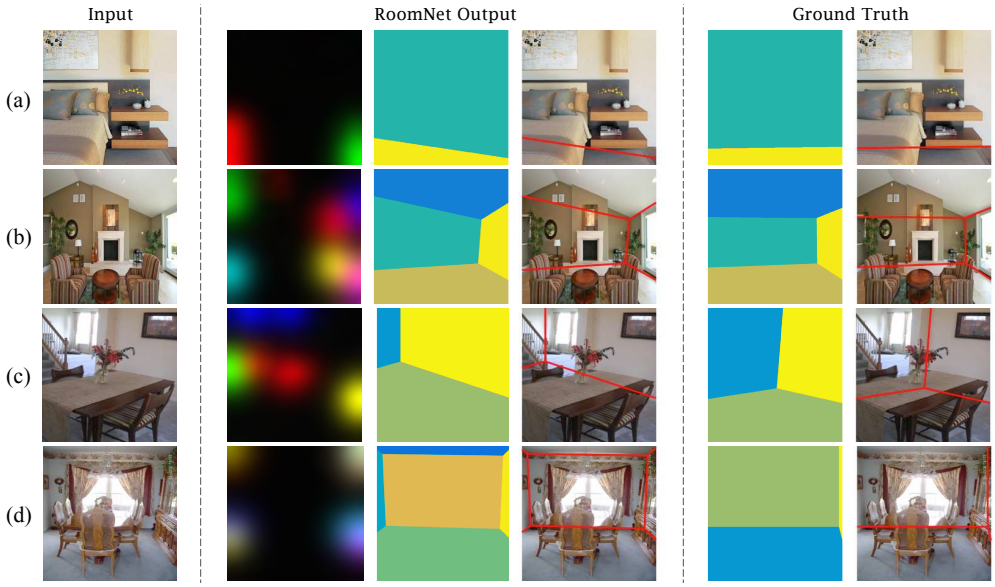

Figure 8. The ambigous cases where the RoomNet predictions do not match the human-annotated ground truth. The first column is the input image, the second column is predicted keypoint heatmaps, the third and fourth columns are obtained boxy representation, and the fifth and sixth columns show the ground truth.

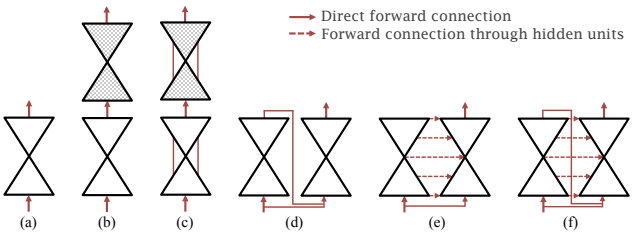

Figure 9. Illustration of different encoder-decoder architecture configurations: (a) vanilla encoder-decoder; (b) stacked encoder-decoder; (c) stacked encoder-decoder with skip-connections; (d) encoder-decoder with feedback; (e) memory augmented recurrent encoder-decoder; (f) memory augmented recurrent encoder-decoder with feedback.

not improve the performance for this task. This could be because the size of the training set (thousands) is not as large as other datasets (millions) that have been evaluated on, therefore skipping layers is not necessary for the specific dataset.

Adding a feedback loop, implemented as a concatenation of input and previous prediction as a new input [44, 33] for the same encoder-decoder network as in (d) improves the performance. At each iteration, the network has access to the thus-far sub-optimal prediction along with the original input to help inference at the current time step.

Making an encoder-decoder recurrent with memory units (e) to behave as a RNN obtains the lowest keypoint error and pixel error (our full RoomNet model). The lateral connections in the recurrent encoder-decoder allow the network to carry information forward and help prediction at future time steps. Finally, adding a feedback loop to the memory

Model	Keypoint Error (%)	Pixel Error (%)
Vanilla enc-dec (RoomNet basic)	6.95	10.46
Stacked enc-dec	6.82	10.31
Stacked enc-dec with skip connect.	7.05	10.48
Enc-dec w/ feedback	6.84	10.10
Recurrent enc-dec (RoomNet full)	**6.30**	**9.86**
Recurrent enc-dec w/ feedback	6.37	9.88

Table 6. Evaluation of encoder-decoder (enc-dec) variants on LSUN dataset [50]. Note that recurrent encoder-decoders use 3 iteration time steps.

augmented recurrent encoder-decoder (f) does not improve the results. It is possible that using the memory augmented structure (e) can already store previous hidden state information well without feedback. Note that weight matrices of the encoder-decoder are not shared in configurations (b) and (c) but shared in configurations (d), (e), and (f), resulting in more parametrically efficient architectures.

5. Conclusion

We presented a simple and direct formulation of room layout estimation as a keypoint localization problem. We showed that our RoomNet architecture and its extensions can be trained end-to-end to perform accurate and efficient room layout estimation. The proposed approach stands out from a large body of work using geometry inspired multi-step processing pipelines. In the future, we would like to adopt gating mechanism [21] to allow incoming signal to alter the state of recurrent units and extend RoomNet to sequential data for building room layout maps

References

[1] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv:1511.00561*, 2015.

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for scene segmentation. *TPAMI*, 2017.

[3] V. Belagiannis and A. Zisserman. Recurrent human pose estimation. *arXiv:1605.02914*, 2016.

[4] J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik. Human pose estimation with iterative error feedback. In *CVPR*, 2016.

[5] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015.

[6] J. M. Coughlan and A. L. Yuille. The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. In *NIPS*, 2000.

[7] S. Dasgupta, K. Fang, K. Chen, and S. Savarese. Delay: Robust spatial layout estimation for cluttered indoor scenes. In *CVPR*, 2016.

[8] L. Del Pero, J. Bowdish, D. Fried, B. Kermgard, E. Hartley, and K. Barnard. Bayesian geometric modeling of indoor scenes. In *CVPR*, 2012.

[9] L. Del Pero, J. Bowdish, B. Kermgard, E. Hartley, and K. Barnard. Understanding bayesian rooms using composite 3d object models. In *CVPR*, 2013.

[10] D. DeTone, T. Malisiewicz, and A. Rabinovich. Deep image homography estimation. *arXiv:1606.03798*, 2016.

[11] A. Gupta, M. Hebert, T. Kanade, and D. M. Blei. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. In *NIPS*, 2010.

[12] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from rgb-d images. In *CVPR*, 2013.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[15] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV*, 2009.

[16] V. Hedau, D. Hoiem, and D. Forsyth. Recovering free space of indoor scenes from a single image. In *CVPR*, 2012.

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 1997.

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

[19] H. Izadinia, Q. Shan, and S. M. Seitz. Im2cad. *arXiv preprint arXiv:1608.05137*, 2016.

[20] 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 MM*, 2014.

[21] C.-Y. Lee, P. W. Gallagher, and Z. Tu. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. In *AISTATS*, 2016.

[22] C.-Y. Lee and S. Osindero. Recursive recurrent nets with attention modeling for ocr in the wild. In *CVPR*, 2016.

[23] C.-Y. Lee, S. Xie, P. W. Gallagher, Z. Zhang, and Z. Tu. Deeply-supervised nets. In *AISTATS*, 2015.

[24] M. Liang and X. Hu. Recurrent convolutional neural network for object recognition. In *CVPR*, 2015.

[25] C. Liu, A. G. Schwing, K. Kundu, R. Urtasun, and S. Fidler. Rent3d: Floor-plan priors for monocular layout estimation. In *CVPR*, 2015.

[26] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016.

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

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

[29] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, et al. Learning to navigate in complex environments. In *ICLR*, 2017.

[30] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

[31] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, 2016.

[32] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *ICCV*, 2015.

[33] M. Oberweger, P. Wohlhart, and V. Lepetit. Training a feedback loop for hand pose estimation. In *ICCV*, 2015.

[34] X. Peng, R. S. Feris, X. Wang, and D. N. Metaxas. A recurrent encoder-decoder network for sequential face alignment. In *ECCV*, 2016.

[35] T. Pfister, J. Charles, and A. Zisserman. Flowing convnets for human pose estimation in videos. In *ICCV*, 2015.

[36] S. Ramalingam, J. K. Pillai, A. Jain, and Y. Taguchi. Manhattan junction catalogue for spatial reasoning of indoor scenes. In *CVPR*, 2013.

[37] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[38] Y. Ren, C. Chen, S. Li, and C.-C. J. Kuo. A coarse-to-fine indoor layout estimation (cfile) method. In *ACCV*, 2016.

[39] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *IJCV*, 2008.

[40] A. G. Schwing, T. Hazan, M. Pollefeys, and R. Urtasun. Efficient structured prediction for 3d indoor scene understanding. In *CVPR*, 2012.

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

[42] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015.

[43] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, 2014.

[44] Z. Tu. Auto-context and its application to high-level vision tasks. In *CVPR*, 2008.

[45] J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, A. Torralba, and W. T. Freeman. Single image 3d interpreter network. In *ECCV*, 2016.

[46] J. Xiao and Y. Furukawa. Reconstructing the worlds museums. *IJCV*, 2014.

[47] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*, 2010.

[48] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *NIPS*, 2015.

[49] J. Zhang, C. Kan, A. G. Schwing, and R. Urtasun. Estimating the 3d layout of indoor scenes and its clutter from depth sensors. In *ICCV*, 2013.

[50] Y. Zhang, F. Yu, S. Song, P. Xu, A. Seff, and J. Xiao. Large-scale scene understanding challenge: Room layout estimation, 2016.

[51] Y. Zhao and S.-C. Zhu. Scene parsing by integrating function, geometry and appearance models. In *CVPR*, 2013.

[52] S. Zheng, S. Jayasumana, B. Romera-Paredes, V. Vineet, Z. Su, D. Du, C. Huang, and P. H. Torr. Conditional random fields as recurrent neural networks. In *CVPR*, 2015.