Tell Me Where I Am: Object-level Scene Context Prediction

Xiaotian Qiao Quanlong Zheng Ying Cao Rynson W.H. Lau
City University of Hong Kong

Abstract

*Contextual information has been shown to be effective in helping solve various image understanding tasks. Previous works have focused on the extraction of contextual information from an image and use it to infer the properties of some object(s) in the image. In this paper, we consider an inverse problem of how to hallucinate missing contextual information from the properties of a few standalone objects. We refer to it as scene context prediction. This problem is difficult as it requires an extensive knowledge of complex and diverse relationships among different objects in natural scenes. We propose a convolutional neural network, which takes as input the properties (*i.e.*, category, shape, and position) of a few standalone objects to predict an object-level scene layout that compactly encodes the semantics and structure of the scene context where the given objects are. Our quantitative experiments and user studies show that our model can generate more plausible scene context than the baseline approach. We demonstrate that our model allows for the synthesis of realistic scene images from just partial scene layouts and internally learns useful features for scene recognition.*

1. Introduction

Scene context refers to how the objects of interest are related to the environment surrounding them. Context information plays an important role in modern computer vision systems. Recent works have leveraged the scene context to improve object detection [15, 24], recognition and segmentation [9, 27, 34, 36], and learn visual feature representation [28]. The prior works attempted to utilize the context information *existed* in an image to infer the properties of some objects of interest in the image. However, an unexplored problem is to predict the *unknown* context of some objects in the image (*i.e.*, to anticipate what and where the missing objects are). Given several foreground objects, humans are remarkably capable of inferring their *unknown* full scene context, by relying on extensive commonsense knowledge of our visual world. For example, given a foreground object in a photo frame as shown in Figure 1, we can reason about multiple plausible environments surrounding it. The properties of the given object provide strong hints about the scene environment, as well as what and where other objects may appear in the scene.

Ying Cao is the corresponding author. This work was led by Rynson Lau.

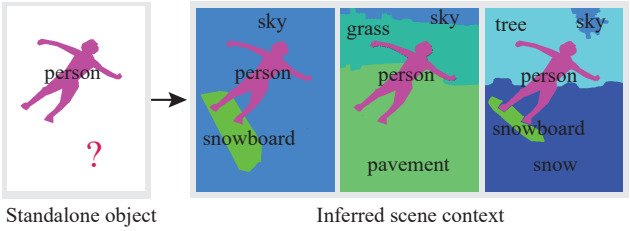

Figure 1. Inferring scene context from a standalone object. A standalone object provides rich information for predicting its scene context (*i.e.*, other objects co-occurring with it and their spatial relations). While the pose and position of the person in the image suggest that the scene may be related to sports activities, the presence and position of the person provide hints as to what and where other objects can appear (*e.g.*, the sky in the upper part of the image, and the pavement in the lower part of the image).

We are thus interested in a fundamental question of whether machines can replicate such scene context inference capability. We believe that the capability of machines to predict where objects are can benefit many scene generation and recognition tasks. However, developing a predictive model for scene context can be challenging, as natural scenes contain a rich variety of semantic objects with complex spatial relations among them. Objects can be at various locations with different scales and shapes. Further, this problem is inherently ambiguous, as the same objects can have multiple semantically plausible scene contexts.

In this paper, for the first time, we address the problem of scene context prediction from standalone objects. To this end, we propose a novel model, which takes as input the categories, shapes, and positions of one or more objects, and predicts the scene context for the given objects. Instead of directly hallucinating low level pixels, our model predicts the context in the form of an object-level scene layout. Although lacking detailed appearances, such a representation captures the important semantics that has been shown to be sufficient to reconstruct photo-realistic images [16]

and build the scene structures [43]. Our model consists of three modules: a shape generator, a region generator, and a compositor. The shape generator aims to generate object shapes of different categories, and the region generator aims to generate object bounding boxes to indicate possible object positions and sizes. The outputs of both generators are then fused and passed to the compositor to generate a scene layout representing the scene context, which is coherent with the input objects.

To evaluate the effectiveness of our method, we conduct quantitative and qualitative experiments on the COCO-Stuff dataset [2]. Experimental results show that our method can generate diverse, semantically plausible scene contexts from the given foreground objects. In addition, we demonstrate that our model enables realistic full scene image synthesis from only partial scene layouts, and that it internally learns useful features for scene recognition.

The main contributions of this paper are:

- To our knowledge, we make the first attempt to address the problem of predicting unknown environments where objects of interest reside in.
- We develop a novel neural network architecture to predict object-level scene contexts from just standalone foreground objects.
- We demonstrate the value of scene context reasoning ability, by showing the utility of our model in both image synthesis and scene recognition tasks.

2. Related Work

Modeling scene context. The context of an image contains rich information about how objects and scenes are related to each other. Cognitive science studies have shown that contextual information plays a crucial role in human visual recognition [1, 7]. There are many types of context information, including visual context [11], global scene context [35], relative location [8], and layout [32].

With deep learning, many tasks are now exploiting contextual information to learn visual features and improve visual understanding performance. On the one hand, context is essential for feature learning. For example, Pathak *et al*. [28] proposed a context encoder to learn high-level semantic features for image inpainting. On the other hand, context has been shown to be effective in many vision tasks, such as recognition, detection and segmentation [27, 36]. Multiple contexts can also be combined to improve performance. Choi *et al*. [6] proposed a graphical model to exploit multiple contexts to identify the out-of-context objects in a scene. Izadinia *et al*. [17] encoded the scene category, the context-specific appearances of objects and their layouts to learn the scene structure. Chien *et al*. [5] built a ConvNet to predict the probability of a pedestrian to be located at some location in the image. Wang *et al*. [40] used a variational auto-encoder to show the possibility of reasonable nonexistent human poses in a scene. All these works use the existing context of the image as an additional cue to reason about the properties of foreground objects of interest. Our objective is fundamentally different from these prior works. Conceptually, we are trying to address an inverse problem, *i.e*., to infer the missing scene context from the properties of the given foreground objects.

Our work also bears some high-level resemblance to the recent efforts on data-driven indoor scene synthesis. They attempted to model object arrangements with undirected factor graphs [19], activity graphs [12], and stochastic grammars [29]. Unlike these works that built context information from pair-wise object relationships, Wang *et al*. [38] introduced a deep neural network to learn the priors of object placements for indoor scene synthesis. Similar to [38] from a high-level perspective, we also use deep neural networks to learn priors on the spatial structure of objects from image data in order to synthesize semantic layouts. However, unlike [38] which aimed to generate the arrangement of a sparse set of 3D objects, we aim to predict a dense, pixel-wise scene layout. In addition, we deal with a more challenging problem as we only use the given object(s) as input but their method assumes the scene types to be known.

Unsupervised representation learning via context prediction. There have been some efforts on unsupervised visual representation learning via context prediction. The skip-gram models [26] learn a word representation by predicting surrounding words of a single word. Doersch *et al*. [10] learned an image representation via predicting the relative position of patches in an image (*i.e*., spatial context). Vondrick *et al*. [37] learned to anticipate the visual representation in a future frame of an unlabeled video (*i.e*., temporal context). In our work, our ultimate goal is not the visual representation learning, but the prediction of the surrounding environment of some standalone objects.

Context-based image manipulation. A number of works have investigated how to use context for image manipulation tasks. Some works used context as a prior to retrieve and composite the assets. Tan *et al*. [33] used CNN features to capture local context for person composition. By jointly encoding the context of foreground objects and background scenes, Zhao *et al*. [44] learned a feature representation for compatible foreground object retrieval based on a given background image. However, the quality of the generated image depends on the retrieval database. The retrieved assets may not fulfill the user's requirement and generate unrealistic compositions. Other works represented the context as a scene layout and learned a generative network to manipulate the synthesized image. Wang *et al*. [39] proposed a GAN model to synthesize and manipulate high resolution images from the scene layout. Hong *et al*. [14] con-

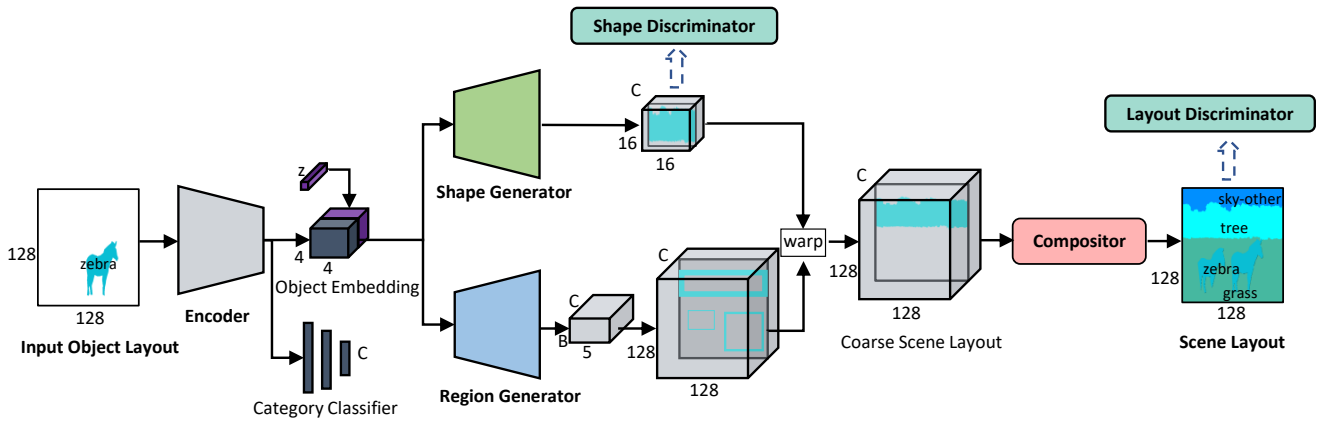

Figure 2. Overview of our network architecture. Our model takes as input an object layout encoding the properties of input objects and generates a scene layout representing the scene context. We use the category classifier to pre-train the encoder to obtain the object embedding features. The object embedding features and a noise vector are concatenated and passed to the shape generator and the region generator. The shape generator generates the shapes of all C object categories, while the region generator generates the parameters and confidence values of B bounding boxes to represent the potential region proposals for each category. The boundary thickness of each bounding box indicates the confidence score of the box. The bounding boxes are then used to warp their corresponding shapes to a coarse scene layout, which is then refined by a compositor to output a final scene layout. Further, a shape discriminator and a layout discriminator are introduced to classify the generated object shapes and scene layouts, respectively, as real or fake.

structed a scene layout as an intermediate representation for image manipulation from text descriptions. A key drawback of these methods is that they need complete semantic layouts or text descriptions as input. With our scene context prediction model, we only need the user to provide a partial semantic layout containing a few objects, and our approach can generate diverse plausible scene layouts for synthesizing realistic full scene images. Thus, our model can be considered to be complementary to existing image synthesis methods.

3. Approach

3.1. Problem Formulation

Our goal is to develop a deep neural network that takes the properties of one or more standalone objects as input to generate a scene context around the objects, which contains the other objects that are likely to co-occur with the given objects. As illustrated in Figure 2, we encode both the input objects and the predicted scene context using the object-level semantic layout, which can compactly describe the classes, shapes, and positions of the objects in a scene layout. Concretely, given an input object layout X_o, our model learns a function f to generate a scene layout $X_s = f(X_o)$. We describe each part of our network in details below.

3.2. Scene Context Prediction Network

Encoder. The input to the encoder is an object layout, $X_o \in \{0,1\}^{H \times W \times C}$, where H and W are the height and width of the layout, respectively, and C indicates the object category of each pixel in the layout in the one-hot vector format. The encoder extracts object embedding features from X_o to produce a feature map of size $4 \times 4 \times 512$. In order to learn a useful object feature representation, we add a category classifier to predict the presence of each object category in the scene context. The category classifier contains two fully connected layers, followed by a Sigmoid layer which outputs a C-dimensional vector.

Shape Generator. To increase the diversity of the generated layout, we add a noise vector z_t on top of the object embedding features by spatial replication and feature channel concatenation, resulting in concatenated features F. We then feed F to the shape generator. The output is the soft binary masks $M \in [0,1]^{16 \times 16 \times C}$, representing the shapes of all object categories. The shape generator module is composed of a sequence of de-convolutional layers. Each layer is a 4×4 de-convolution with a stride of 2, followed by batch normalization and ReLU. The last layer is a 1×1 convolution followed by Sigmoid nonlinearity.

Region Generator. Region generator receives F and predicts B region proposals for each of the C object categories. Each region proposal is represented by a bounding box with four parameters (x, y, w, h) and a confidence score s. (x, y) refer to the center location of the box. (w, h) refer to the width and height of the box. The confidence value represents the probability that the bounding box covers an object. Thus, the output of the region generator is a tensor of size $B \times 5 \times C$, where 5 refers to the four parameters plus the confidence score. The object generator module consists of a set of residual blocks and convolution operations, fol-

lowed by two fully connected layers to predict the bounding box parameters and the confidence scores.

Compositor. To combine the predicted shape masks and object bounding boxes coherently into a scene layout, for each object category, we warp the generated shape masks to the position of the corresponding bounding box using the bilinear interpolation operator in the spatial transformer network [18]. Note that some artifacts like unlabeled regions and tiny objects may exist in the fused coarse scene layout. In addition, the generated object bounding boxes may overlap each other, causing occlusion among different objects. To address these problems, we further convert the coarse layout to the dense pixelwise scene layout using the Cascaded Refinement Module [3].

Discriminators. For a given input (*e.g.*, a standalone object on a canvas), there may be multiple scene layouts that are plausible and consistent with the input. To handle this multi-modal issue, we introduce two additional discriminators, as inspired by the recent success of adversarial learning approaches [13, 16]. One is a shape discriminator D_{shape}, and the other is a layout discriminator D_{layout}. The input to the shape discriminator D_{shape} is the generated shape masks $m_c^{'}$ or the real shape masks m_c. The input to the layout discriminator D_{layout} is the generated scene layouts or the real scene layouts. Both discriminators encode the input by a series of downsampling layers, which are implemented by stride-2 convolutions.

3.3. Training

Due to the complexity of context prediction, it is difficult to directly train our model end-to-end. Thus, we first pre-train the category classifier to obtain the object embedding features. We then train all modules together. Let $l = \{l_c \in \{0,1\}, c \in C\}$ be the ground truth object categories of an image. We use the cross-entropy loss L_{cls} for the category classifier as:

$$L_{cls} = -\sum_{c \in C} [l_c \log p_c + (1 - l_c) \log(1 - p_c)], \quad (1)$$

where $l_c = 1$ if the image contains object category c. p_c is the predicted probability for c.

For the region generator, we use $t = (x, y, \sqrt{w}, \sqrt{h})^T$ and s to denote the bounding box parameters and the confidence score of a predicted bounding box, respectively. We define the loss as:

$$L_{box} = \sum_{i=1}^{C} \sum_{j \in B_i} (p_{i,j}^{obj} ||t_{i,j} - \hat{t}_{i,j}||^2 + \lambda(p_{i,j}^{obj}) ||s_{i,j} - \hat{s}_{i,j}||^2), \quad (2)$$

where i ranges over all object categories and j ranges over all the bounding boxes of category i. $\hat{t}_{i,j}$ and $\hat{s}_{i,j}$ are the ground truth bounding box parameters and the confidence score, respectively. $p_{i,j}^{obj}$ is an object indicator that is 1 when the bounding box covers a ground truth object and 0 otherwise. Since most generated bounding boxes do not cover any ground truth objects, we introduce a class re-balancing function $\lambda(x)$ to prevent the model from predicting a zero confidence score for most bounding boxes. $\lambda(x) = 1$ when $x = 1$ and $\lambda(x) = 0.1$ when $x = 0$.

For the shape generator, let $m \sim p_{fake}(m)$ be the generated shapes. Its loss is defined as:

$$\begin{aligned} L_{shape} =& E_{m \sim p_{fake}(m)}[L_{crs}(m, \hat{m})] + \\ & E_{m \sim p_{fake}(m)}[(D_{shape}(m) - 1)^2]. \end{aligned} \quad (3)$$

The first term penalizes the difference between each generated shape m and its ground truth $\hat{m}$ using a pixelwise cross-entropy loss L_{crs}. The second term encourages the shape generator to produce realistic shapes to fool the shape discriminator. We use L2 norm instead of $\log$, as in the LSGAN [25], to stabilize our training.

For the shape discriminator, its adversarial loss is defined as:

$$\begin{aligned} L_{shape}^{D} =& E_{m \sim p_{fake}(m)}[(D_{shape}(m) - 1)^2] + \\ & E_{t \sim p_{real}(t)}[D_{shape}(t)^2], \end{aligned} \quad (4)$$

where $t \sim p_{real}(t)$ are real shapes.

Finally, the losses for the output scene layout and the layout discriminator are identical to Eq. 3 and 4, respectively, except that object shapes are replaced with scene layouts.

3.4. Implementation Details

The input object layout is resized to 128×128 by nearest interpolation. To help the network converge, we first train the category classifier to obtain the object embedding features. To obtain the values of the object indicator $p_{i,j}^{obj}$, which are used in the loss in Eq. 2, we follow the approach of YOLO [31]. In particular, for each iteration during training, we feed an input into our network to predict the region proposals for all object categories. A region proposal of a category is labeled as positive (*i.e.*, $p_{i,j}^{obj} = 1$) only if it has an intersection over union (IOU) of at least 0.5 with any ground truth bounding box of the same category, and labeled as negative (*i.e.*, $p_{i,j}^{obj} = 0$) otherwise. The ground truth confidence score $\hat{s}_{i,j}$ of a predicted bounding box is defined as the IOU between the predicted bounding box and the ground truth. Note that during the training stage, we use the ground truth of object bounding boxes for the compositor to generate the scene layout. During the testing stage, we choose the generated object bounding boxes based on the corresponding confidence value. We use the Adam optimizer [20] for optimization with $\beta_1 = 0.5$ and $\beta_2 = 0.9999$. The learning rate is $2e^{-4}$ and the batch size is 128. For each mini-batch, we alternately minimize the loss functions to update the parameters of the generators and discriminators, as in [13].

4. Experiments

In this section, we train our model to generate scene layouts on the COCO-Stuff [2] dataset. We aim to show that our model can generate plausible scene contexts from the input objects with diverse object properties. We show both qualitative and quantitative results of our method, in comparison with a baseline, and evaluate the plausibility of our generated scene contexts via a user study. Finally, we show how our model can be used to synthesize realistic scene images from partial scene layouts and help performance improvement on scene recognition.

4.1. Dataset

We perform experiments on the COCO-Stuff dataset, which augments a subset of the COCO dataset [22] with additional stuff categories. The dataset annotates 40k training and 5k testing images with bounding boxes and semantic layouts for both indoor and outdoor scenes, including 80 *thing* categories and 91 *thing* categories in total. Our evaluation only focuses on outdoor scene images in the dataset, which have many complex and diverse scene structures and thus make the scene context prediction problem very challenging. Given the outdoor scene layouts, we only select those with 2 to 8 objects in them. To train our network, for each layout, we randomly select one or two objects and remove other regions to form an *object layout*, which together with the original semantic layout (referred to as *scene layout* in this work) form a training pair. If a selected object covers less than 5% of the image, we skip this object. As a result, we produce a dataset that contains 72 object categories (39 *thing* categories and 33 *stuff* categories), with a total of 52,803 training pairs and 1,934 test pairs.

4.2. Baseline

Since we are not aware of any previous works on scene context prediction, we compare our method with the pix2pix method [16], which learns a generic mapping between an input image and an output image with aligned image pairs. We train the baseline using our training dataset so that it can map an input object layout to an output scene layout, as in our model. Note that we have empirically found that the baseline tends to produce noise, *i.e.*, small artifacts, in its outputs. Thus, we refine its results via a simple post-processing to address the issue. Specifically, we first filter the initial results by a weighted median filter. We then apply a guided filter with the filtered images as guide images to obtain the final results.

4.3. Qualitative Results

Figure 3 shows some qualitative results of our model, in comparison with those from the baseline. We make several observations as follows. First, our method can generate more visually diverse scene layouts than the baseline. For example, in the first column, the baseline always gives similar object categories and positions (*e.g.*, grass, tree and sky), while our method can generate diverse object categories and positions, in response to the input object layouts. Second, the scene layouts predicted by our model are more semantically plausible than those by the baseline. For example, in the third row of the first column, the baseline predicts some unlikely spatial relations among the objects for the scene (*i.e.*, person in sky and grass in sky). In contrast, our method predicts a sea region given the bird flying on top of the sand, which is more convincing. Third, our model is able to generate scene context that respects the input objects consistently, while the baseline fails to give reasonable results in some cases (*e.g.*, landing the snowboard on top of the grass in the fourth row of the first column).

We further investigate how well the change in the spatial relationship between the input objects may affect the scene context inference of our model, and show the results in the second column. For example, when we change the spatial relationship of the person and the airplane from left/right (first row) to above/below (second row) or inside/surrounding (fourth row), the scene layouts by our model favorably adapt to the inputs, while the baseline tends to give similar results. We also analyze how different object sizes and categories affect the predicted scene layouts. In the third column, we show the different generated scene layouts when changing the size of a car (first and second rows) or changing the category of the input from boat to tennis racket (third and fourth rows). Our method can still give promising results, whereas the baseline predicts less plausible context sometimes (*e.g.*, putting a car on grass (second row)). These results suggest that, as compared with the baseline, our method can generate more diverse, semantically plausible scenes that fit the input object layouts.

4.4. User Studies

We use Amazon Mechanical Turk (AMT) to evaluate the quality of our results in terms of two aspects: plausibility (how plausible the generated scene layouts are) and fitness (how well the generated scene layouts fit the given inputs). For this experiment, we use 50 input object layouts randomly chosen form our test dataset. For each input, we generate scene layouts using our method (Ours) and pix2pix (Baseline).

Plausibility. We ask AMT workers to judge the generated layouts and ground truth (GT), by evaluating whether objects in the scene layouts have incorrect relationships (plausibility). For those scene layouts that are regarded as implausible, they are asked to label at least a pair of objects that have a wrong spatial relation. We had a total of 30 workers in our experiment, with each scene layout being evaluated by at least 10 workers.

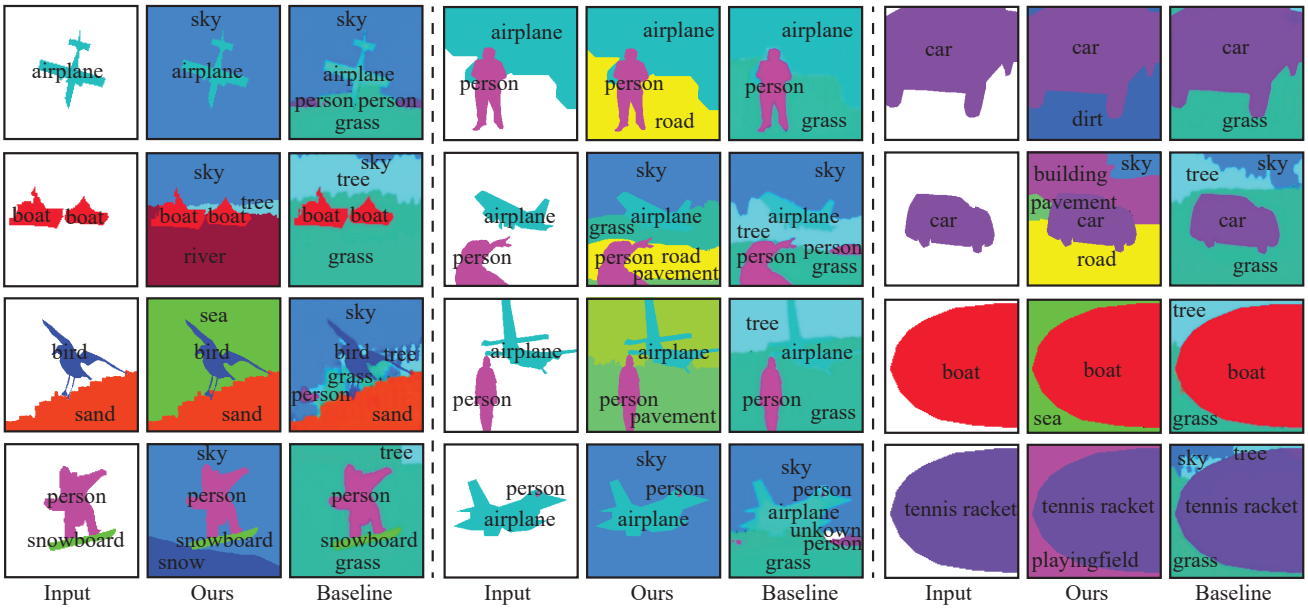

Figure 3. Qualitative results from our model and the baseline. Given the input object layouts (left diagrams in each column), which contains one or two standalone objects, we generate the output scene layouts using our model (middle diagrams in each column) and the baseline (right diagrams in each column).

	Baseline	Ours	GT
Plausibility score ↑	0.34 ± 0.12	0.73 ± 0.11	0.85 ± 0.08

Table 1. Plausibility scores for the baseline (Baseline), our method (Ours), and the ground truth (GT).

	Ours over Baseline	Ours over GT	Baseline over GT
Fitness preference	71%	43%	16%

Table 2. Fitness preference for the baseline (Baseline), our method (Ours), and the ground truth (GT).

For each scene layout, we compute the fraction of workers who have chosen it to be plausible as a plausibility score, and report the average score for each method in Table 1. Note that the average score of the ground truth represents an upper bound performance. The results show that our results are perceived to be significantly plausible than those by the baseline and much closer to the ground truth.

Fitness. For this experiment, we show the workers an input object layout along with two scene layouts that are randomly chosen from the three sources (Ours, Baseline and GT). The workers are asked to select a scene layout that describes a more plausible context of the input objects. We had a total of 9 workers in the experiment. The results in Table 2 show that our results are preferred by the workers most of time, compared with those by the baseline. Further, the workers only show a slight preference for the ground truth layouts over our results. This implies that our results are considered to fit the input objects better, and comparable to the ground truth.

4.5. Quantitative Evaluation

We quantitatively evaluate the plausibility of our generated scene layouts using the object pairwise relationship priors, which have been widely used in indoor scene synthesis to characterize scene structure [23, 38]. In particular, we compute the probabilities of pairwise relations among object classes from a dataset of natural scene images, and evaluate the likelihood of each generated scene layout under the probabilities to measure its quality. Formally, let C and R be the object categories and pairwise spatial relations, respectively. For each pair of object classes $\langle u, v \rangle, u, v \in C$, we compute a probability of them being in a spatial relation $r \in R$ as $p(r|\langle u, v \rangle)$. Here, we consider six mutually exclusive spatial relationships, ie, $R \in \{left, right, above, below, inside, outside\}$. Given a generated scene layout X, we define its negative log likelihood (NLL) as:

$$NLL = -\frac{\sum_{\langle u,r,v \rangle \in T} \log p(r|\langle u, v \rangle)}{|T|}, \quad (5)$$

where $\langle u, r, v \rangle$ iterates over all the possible class pairs denoted as T in the layout.

We use 2-fold cross-validation for this evaluation. In particular, we first split our training dataset uniformly into two fold. For each fold, we train a model on it, learn the priors from the other fold, and compute the NLL on the test dataset against the priors. Finally, we use the mean NLL (NLL_{all}) over the two fold as our metric for scene layout plausibility evaluation. In addition, we calculate the input-centric

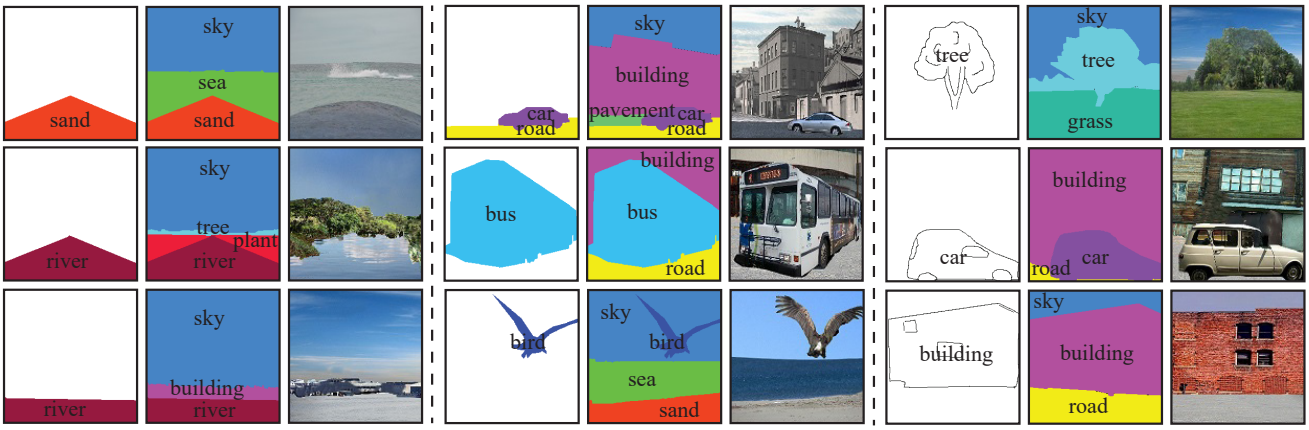

Figure 4. Given a partial scene layout or a sketch as input, our method is able to generate a complete scene layout and further synthesize realistic a full scene image.

mean NLL (NLL_{object}) to measure how well the predicted scene layouts fit the inputs. To do this, we only consider the category pairs where the inputs are involved in Eq. 5.

Table 3 compares the performance of our model to the baseline. Our method outperforms the baseline by a large margin in both metrics. This again confirms the superior performance of our method in predicting plausible and fitting scene context, in comparison to the baseline.

4.6. Ablation Study

To investigate how different components in our network affect the generation performance, we compare several ablated versions of our model, using the mean NLL introduced in Section 4.5.

No category classifier. We remove the category classifier, so that there is no pre-training for the object embedding features.

No discriminators. We remove both shape and layout discriminators, relying only on the pixelwise cross entropy losses for model learning.

No shape or layout discriminator. We remove one of the discriminators.

From the results in Table 3, we can observe that compared with the ablated versions, our full model achieves better performance, which demonstrates the necessity of each component in our model.

4.7. Image Synthesis

We conduct experiments by using our model for image synthesis. Several recent promising works [3, 30, 39] on image synthesis have attempted to generate realistic images from scene layouts. While being able to synthesize stunning results, they all need a complete scene layout to start with. The ability of our model to infer scene context from only standalone foreground objects makes it possible to hallucinate a *full* scene image with just a *partial* semantic layout.

Method	$NLL_{all} \downarrow$	$NLL_{object} \downarrow$
Baseline [16]	2.15	2.11
Ours (No category classifier)	1.77	1.70
Ours (No discriminators)	1.91	1.85
Ours (No shape discriminator)	1.72	1.61
Ours (No layout discriminator)	1.88	1.84
Ours (Full model)	**1.69**	**1.57**

Table 3. Quantitative evaluation of the baseline, the ablated versions of our model and our full model. We evaluate the performance with negative log likelihood (NLL) of the generated layouts under pre-computed pairwise relation priors. NLL_{all} reflects the overall plausibility of an output layout, and NLL_{object} indicates the fitness between the input objects and an output layout.

For this task, we leverage the state-of-art image synthesis method [30], which transforms a semantic layout into a realistic image. The segments are extracted from our training dataset to generate the memory bank. Given a partial scene layout, we first use our model to predict a full scene layout, which is then fed into the image synthesis method to produce an output image.

In addition to partial semantic layout, we also experiment with using a sketch as an input to our model for image synthesis. To do this, we first need to convert a sketch into a partial layout required by our model. In particular, given the images and their semantic layouts in our training dataset, we use edge detection [42] to obtain the sketches of randomly chosen foreground objects, by applying post-processing steps as in [4] (including binarization, thinning, small component removal, erosion and spur removal). After that, we train a pix2pix network [16] to map the sketches to

Method	Accuracy
Chance	0.5%
ImageNet-CNN [21]	38.9%
Places-CNN [46]	49.8%
Ours + SVM	39.8%
Ours + Random Init	37.6%
Ours + Finetune	**52.4%**

Table 4. Outdoor scene recognition accuracy on the SUN dataset [41]. We evaluate the representation learned by our layout discriminator for scene recognition. We compare the performance of directly using the learned representation with a SVM (Ours + SVM), randomly initializing the discriminator (Ours + Random Init) and fine-tuning the discriminator (Ours + Finetune). For comparison, we also show the results from ImageNet-CNN and Places-CNN.

partial layouts. During the test stage, given an input sketch, we first map it to a partial layout, and then transform it to a full scene image with the above image synthesis process.

In Figure 4, we show some image synthesis results generated from partial semantic layouts and sketches. As can be seen, our method can synthesize complex and semantically meaningful full scene images from sparse user inputs.

4.8. Scene Recognition

We also test the the representation learned by the layout discriminator for outdoor scene recognition on the SUN dataset [41]. Note that we use 220 outdoor scene categories for evaluation since our model is trained on outdoor scenes.

To do this, we first replace the output layer of our discriminator with a K-way softmax layer. We then construct a scene recognition model by using a pre-trained semantic segmentation model [45] to map an input color image to a scene layout, which is then fed into our discriminator for classification. We fine-tune our discriminator using the training splits of the SUN dataset (Ours + Finetune). We also experiment with randomly initializing the discriminator part of our recognition model (Ours + Random Init) instead of using learned weights of our discriminator, and directly using the outputs of the penultimate layer of the discriminator as features for a multi-class SVM (Ours + SVM). Note that since we are interested in exploring the representation of our discriminator, we fix the weights of the semantic segmentation model during the experiment.

We report the recognition accuracy in Table 4. SVM using our learned representation as features slightly outperforms AlexNet pre-trained on ImageNet [21], but is inferior to the pre-trained Places-CNN [46] that is especially designed for scene recognition. In addition, while our randomly initialized model is worse than ImageNet-CNN and Places-CNN, the model initialized from the weights of our learned discriminator obtains better performance. This is possibly because, in order to discriminate between real and fake scene layouts, our discriminator needs to learn a representation that captures complex semantic and spatial relationships among the objects in a scene layout, which is important to excel scene recognition performance. These results suggest that learning to hallucinate object-level scene context helps learn useful features for scene recognition.

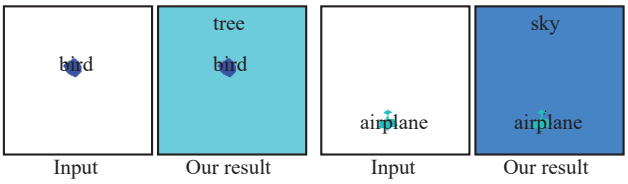

Figure 5. Failure cases. When the size of an input object is too small, our method may generate an over-simplified scene context with a large background only.

5. Conclusion

In this paper, we make an effort to address the problem of reasoning about the missing environment from the properties of a few standalone objects. To this end, we propose a scene context prediction model that estimates scene layouts from input object layouts in an end-to-end manner. Extensive qualitative and quantitative results show that our model is able to generate more plausible and diverse scene layouts that put the input objects into the right context, as compared with a baseline model. In addition, we demonstrate that the ability to predict scene contexts enables a image synthesis approach that can generate full scene images from only sparse, partial user inputs. Finally, we show that learning to hallucinate scene contexts can be a promising supervisory signal for learning useful features for scene recognition.

Currently, our model may fail if the input objects are too small. As shown in Figure 5, if the input layout contains only a small object, it tends to produce an over-simplified scene layout with a large background, even though the synthesized scene layout is still plausible. To address this issue, we would like to explore a multi-scale object modeling approach to deal with small objects in our future work.

References

[1] M. Bar. Visual objects in context. *Nature Reviews Neuroscience*, 5(8):617, 2004. 2

[2] H. Caesar, J. Uijlings, and V. Ferrari. Coco-stuff: Thing and stuff classes in context. In *CVPR*, 2018. 2, 5

[3] Q. Chen and V. Koltun. Photographic image synthesis with cascaded refinement networks. In *ICCV*, 2017. 4, 7

[4] W. Chen and J. Hays. Sketchygan: Towards diverse and realistic sketch to image synthesis. In *CVPR*, 2018. 7

[5] J.-T. Chien, C.-J. Chou, D.-J. Chen, and H.-T. Chen. Detecting nonexistent pedestrians. In *ICCVW*, 2017. 2

[6] M. J. Choi, A. Torralba, and A. S. Willsky. Context models and out-of-context objects. *Pattern Recognition Letters*, 33(7):853–862, 2012. 2

[7] M. M. Chun and Y. Jiang. Contextual cueing: Implicit learning and memory of visual context guides spatial attention. *Cognitive psychology*, 36(1):28–71, 1998. 2

[8] C. Desai, D. Ramanan, and C. C. Fowlkes. Discriminative models for multi-class object layout. *IJCV*, 95(1):1–12, 2011. 2

[9] S. K. Divvala, D. Hoiem, J. H. Hays, A. A. Efros, and M. Hebert. An empirical study of context in object detection. In *CVPR*, 2009. 1

[10] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *ICCV*, 2015. 2

[11] N. Dvornik, J. Mairal, and C. Schmid. Modeling visual context is key to augmenting object detection datasets. In *ECCV*, 2018. 2

[12] Q. Fu, X. Chen, X. Wang, S. Wen, B. Zhou, and H. Fu. Adaptive synthesis of indoor scenes via activity-associated object relation graphs. *ACM TOG*, 36(6):201, 2017. 2

[13] 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. 4

[14] S. Hong, D. Yang, J. Choi, and H. Lee. Inferring semantic layout for hierarchical text-to-image synthesis. In *CVPR*, 2018. 2

[15] H. Hu, J. Gu, Z. Zhang, J. Dai, and Y. Wei. Relation networks for object detection. In *CVPR*, 2018. 1

[16] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 1, 4, 5, 7

[17] H. Izadinia, F. Sadeghi, and A. Farhadi. Incorporating scene context and object layout into appearance modeling. In *CVPR*, 2014. 2

[18] M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *NIPS*, 2015. 4

[19] Z. S. Kermani, Z. Liao, P. Tan, and H. Zhang. Learning 3d scene synthesis from annotated rgb-d images. In *Computer Graphics Forum*, volume 35, pages 197–206. Wiley Online Library, 2016. 2

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. 4

[21] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 8

[22] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 5

[23] T. Liu, S. Chaudhuri, V. G. Kim, Q. Huang, N. J. Mitra, and T. Funkhouser. Creating consistent scene graphs using a probabilistic grammar. *ACM TOG*, 2014. 6

[24] Y. Liu, R. Wang, S. Shan, and X. Chen. Structure inference net: Object detection using scene-level context and instance-level relationships. In *CVPR*, 2018. 1

[25] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley. Least squares generative adversarial networks. In *ICCV*, 2017. 4

[26] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013. 2

[27] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille. The role of context for object detection and semantic segmentation in the wild. In *CVPR*, 2014. 1, 2

[28] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 1, 2

[29] S. Qi, Y. Zhu, S. Huang, C. Jiang, and S.-C. Zhu. Human-centric indoor scene synthesis using stochastic grammar. In *CVPR*, 2018. 2

[30] X. Qi, Q. Chen, J. Jia, and V. Koltun. Semi-parametric image synthesis. In *CVPR*, 2018. 7

[31] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016. 4

[32] A. G. Schwing, S. Fidler, M. Pollefeys, and R. Urtasun. Box in the box: Joint 3d layout and object reasoning from single images. In *ICCV*, 2013. 2

[33] F. Tan, C. Bernier, B. Cohen, V. Ordonez, and C. Barnes. Where and who? automatic semantic-aware person composition. In *WACV*, 2018. 2

[34] A. Torralba. Contextual priming for object detection. *IJCV*, 53(2):169–191, 2003. 1

[35] A. Torralba, K. Murphy, and W. Freeman. Using the forest to see the trees: Object recognition in context. *Comm. of the ACM*, 2010. 2

[36] Z. Tu and X. Bai. Auto-context and its application to high-level vision tasks and 3d brain image segmentation. *IEEE TPAMI*, 32(10):1744–1757, 2010. 1, 2

[37] C. Vondrick, H. Pirsiavash, and A. Torralba. Anticipating the future by watching unlabeled video. 2016. 2

[38] K. Wang, M. Savva, A. X. Chang, and D. Ritchie. Deep convolutional priors for indoor scene synthesis. *ACM TOG*, 37(4):70, 2018. 2, 6

[39] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 2, 7

[40] X. Wang, R. Girdhar, and A. Gupta. Binge watching: Scaling affordance learning from sitcoms. In *CVPR*, 2017. 2

[41] 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. 8

[42] S. Xie and Z. Tu. Holistically-nested edge detection. In *ICCV*, 2015. 7

[43] D. Xu, Y. Zhu, C. B. Choy, and L. Fei-Fei. Scene graph generation by iterative message passing. In *CVPR*, 2017. 2

[44] H. Zhao, X. Shen, Z. Lin, K. Sunkavalli, B. Price, and J. Jia. Compositing-aware image search. In *ECCV*, 2018. 2

[45] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 8

[46] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014. 8