SG-VAE: Scene Grammar Variational Autoencoder to Generate New Indoor Scenes

Downloaded from: https://research.chalmers.se, 2026-10-07 04:10 UTC

Citation for the original published paper (version of record):

Purkait, P., Zach, C., Reid, I. (2020). SG-VAE: Scene Grammar Variational Autoencoder to Generate New Indoor Scenes. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 12369 LNCS: 155-171. http://dx.doi.org/10.1007/978-3-030-58586-0_10

N.B. When citing this work, cite the original published paper.

research.chalmers.se offers the possibility of retrieving research publications produced at Chalmers University of Technology. It covers all kind of research output: articles, dissertations, conference papers, reports etc. since 2004. research.chalmers.se is administrated and maintained by Chalmers Library

(article starts on next page)

SG-VAE: Scene Grammar Variational Autoencoder to Generate New Indoor Scenes

Pulak Purkait[1(✉)], Christopher Zach[2], and Ian Reid[1]

[1] Australian Institute of Machine Learning and School of Computer Science, The University of Adelaide, Adelaide, SA 5005, Australia
pulak.isi@gmail.com

[2] Chalmers University of Technology, 41296 Goteborg, Sweden

Abstract. Deep generative models have been used in recent years to learn coherent latent representations in order to synthesize high-quality images. In this work, we propose a neural network to learn a generative model for sampling consistent indoor scene layouts. Our method learns the co-occurrences, and appearance parameters such as shape and pose, for different objects categories through a grammar-based auto-encoder, resulting in a compact and accurate representation for scene layouts. In contrast to existing grammar-based methods with a user-specified grammar, we construct the grammar automatically by extracting a set of production rules on reasoning about object co-occurrences in training data. The extracted grammar is able to represent a scene by an augmented parse tree. The proposed auto-encoder encodes these parse trees to a latent code, and decodes the latent code to a parse tree, thereby ensuring the generated scene is always valid. We experimentally demonstrate that the proposed auto-encoder learns not only to generate valid scenes (i.e. the arrangements and appearances of objects), but it also learns coherent latent representations where nearby latent samples decode to similar scene outputs. The obtained generative model is applicable to several computer vision tasks such as 3D pose and layout estimation from RGB-D data.

Keywords: Scene grammar · Indoor scene synthesis · VAE

1 Introduction

Recently proposed approaches for deep generative models have seen great success in producing high quality RGB images [7,16,17,24] and continuous latent representations from images [12]. Our work aims to learn coherent latent representations for generating natural indoor scenes comprising different object categories

Electronic supplementary material The online version of this chapter (https://doi.org/10.1007/978-3-030-58586-0_10) contains supplementary material, which is available to authorized users.

© Springer Nature Switzerland AG 2020
A. Vedaldi et al. (Eds.): ECCV 2020, LNCS 12369, pp. 155–171, 2020.
https://doi.org/10.1007/978-3-030-58586-0_10

and their respective appearances (*i.e.* pose and shape). Such a learned representation has direct use for various computer vision and scene understanding tasks, including (**i**) 3D scene-layout estimation [27], (**ii**) 3D visual grounding [3,31], (**iii**) Visual Question Answering [1,18], and (**iv**) robot navigation [19].

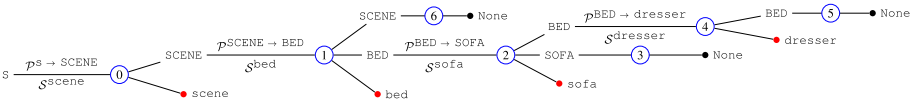


Fig. 1. An example of parse tree obtained by applying the CFG to a scene comprising `bed, sofa, dresser`. The sequence of production rules ⓪–⑥ are marked in order. The attributes of production rules are displayed above and below of the rules.

Developing generative models for such discrete domains has been explored in a limited number of works [9,23,34]. These works utilize prior knowledge of indoor scenes by manually defining attributed grammars. However, the number of rules in such grammars can be prohibitively large for real indoor environments, and consequently, these methods are evaluated only on synthetic data with a small number of objects. Further, the Monte Carlo based inference method can be intractably slow: up to 40 min [23] or one hour [9] to estimate a single layout. Deep generative models for discrete domains have been proposed in [6] (employing sequential representations) and in [14] (based on formal grammars). In our work, we extend [14] by integrating object attributes, such as pose and shape of objects in a scene. Further, the underlying grammar is often defined manually [14,23], but we propose to extract suitable grammar rules from training data automatically. The main components of our approach are thus:

- a scene grammar variational autoencoder (SG-VAE) that captures the appearances (*i.e.* pose and shape) of objects in the same 3D spatial configurations in a compact latent code (Sect. 2);
- a context free grammar that explains causal relationships among objects which frequently co-occur, automatically extracted from training data (Sect. 3);
- the practicality of the learned latent space is also demonstrated for a computer vision task.

Our SG-VAE is fast and has the ability to represent the scene in a coherent latent space as shown in Sect. 4.

2 Deep Generative Model for Scene Generation

The proposed method is influenced by the Grammar Variational Autoencoder [14], so we begin with a brief description of that prior art.

The Grammar VAE takes a valid string (in their case a chemical formula) and begins by parsing it into a set of production rules. These rules are represented as

1-hot binary vectors and encoded compactly to a latent code by the VAE. Latent codes can then be sampled and decoded to production rules and corresponding valid strings. More specifically, each production rule is represented by a 1-hot vector of size N, where N is the total number of rules, *i.e.* $N = |\mathcal{R}|$ (where $\mathcal{R}$ is the set of rules). The maximum size T of the sequence is fixed in advance. Thus the scene is represented by a sequence $\mathcal{X} \in \{0,1\}^{N \times T}$ of 1-hot vectors (note that when fewer than T rules are needed, a dummy/null rule is used to pad the sequence up to length T ensuring that the input to the autoencoder is always the same size). $\mathcal{X}$ is then encoded to a continuous (low)-dimensional latent posterior distribution $\mathcal{N}(\boldsymbol{\mu}(\mathcal{X}), \boldsymbol{\Sigma}(\mathcal{X}))$. The decoding network, which is a recurrent network, maps latent vectors to a set of unnormalized log probability vectors (logits) corresponding to the production rules. To convert from the output logits to a valid sequence of production rules, each logit vector is considered in turn. The max output in the logit vector gives a 1-hot encoding of a production rule, but only some sequences of rules are valid. To avoid generating a rule that is inconsistent with the rules that have preceded it, invalid rules are masked out of the logit and the max is taken over only unmasked elements. This ensures that the Grammar VAE only ever generates valid outputs. Further details of the Grammar VAE can be found in [14].

Adapting this idea to the case of generating scenes requires that we incorporate not only valid co-occurrences of objects, but also valid attributes such as absolute pose (3D location and orientation) and shape (3D bounding boxes) of the objects in the scene. More specifically, our proposed SG-VAE is adapted from the Grammar VAE in the following ways:

- The object attributes, *i.e.* absolute pose and shape of the objects are estimated while inferring the production rules.
- The SG-VAE is moreover designed to generate valid 3D scenes which adhere not only to the rules of grammar, but also generate valid poses.

2.1 Scene-Grammar Variational Autoencoder

We represent the objects in indoor scenes explicitly by a set of production rules, so that the entire arrangement—*i.e.* the occurrences and appearances (*i.e.* pose and shape) of the objects in a scene—is guaranteed to be consistent during inference. Nevertheless we also aim to capture the advantages of deep generative models in admitting a compact representation that can be rapidly decoded. While a standard VAE would implicitly *encourage* decoded outputs to be scene-like, our proposed solution extends the Grammar VAE [14] to explicitly enforce an underlying grammar, while still possessing the aforementioned advantages of deep generative models. For example, given an appearance of an object *bed*, the model finds strong evidence for co-occurrence of another indoor object, *e.g. dresser*. Furthermore, given the attributes (3D pose and bounding boxes) of one object (*bed*), the attributes of the latter (*dresser*) can be inferred.

The model comprises two parts: **(i)** a context free grammar (CFG) that represents valid configurations of objects; **(ii)** a Variational Autoencoder (VAE)

that maps a sequence of production rules (*i.e.* a valid scene) to a low dimensional latent space, and decodes a latent vector to a sequence of production rules which in turn define a valid scene.

2.2 CFG of Indoor Scenes

A context-free grammar can be defined by a 4-tuple of sets $G = (S, \Sigma, \mathcal{V}, \mathcal{R})$ where S is a distinct non-terminal symbol known as start symbol; Σ is the finite set of non-terminal symbols; $\mathcal{V}$ is the set of terminal symbols; and $\mathcal{R}$ is the set of production rules. Note that in a CFG, the left hand side is always a non-terminal symbol. A set of all valid configurations $\mathcal{C}$ derived from the production rules defined by the CFG G is called a language. In contrast to [23] where the grammar is pre-specified, we propose a data-driven algorithm to generate a set of production rules that constitutes a CFG.

We select a few objects and associate a number of non-terminals. Only those objects that lead to co-occurrence of other objects also exist as non-terminals (described in detail in Sect. 3.2). A valid production rule is thus *"an object category, corresponding to a non-terminal, generates another object category"*. For clarity, non-terminals are denoted in upper-case with the object name. For example `BED` and `bed` are the non-terminal and the terminal symbols corresponding to the object category *bed*. Thus occurrence of a non-terminal `BED` leads to occurrence of the immediate terminal symbol `bed` and possibly further occurrences of other terminal symbols that *bed* co-occurs with, *e.g.* `dresser`. Thus, a set of rules {`S` → `scene SCENE;` `SCENE` → `bed BED SCENE;` `BED` → `bed BED;` `BED` → `dresser BED;` `BED` → `None;` `SCENE` → `None`} can be defined accordingly. Note that an additional object category *scene* is incorporated to represent the shape and size of the room. The learned scene grammar is composed of following rules:

(R1) *involving start symbol* `S`: generates the terminal `scene` and non-terminal `SCENE` that represents the indoor scene layout with attributes as the room size and room orientation, *e.g.* `S` → `scene SCENE;`. This rule ensures generating a room first.
(R2) *involving non-terminal* `SCENE`: generates a terminal and a non-terminal corresponding to an object category, *e.g.* `SCENE` → `bed BED SCENE;`.
(R3) *generating a terminal object category*: a non-terminal generates a terminal corresponding to another object category, *e.g.* `BED` → `dresser BED;`.
(R4) *involving* `None`: non-terminal symbols assigned to `None`, *e.g.* `BED` → `None;`.

`None` is an empty object and corresponding rule is a dummy rule indicating that the generation of the non-terminal is complete and the parser is now ready to handle the next non-terminal in the stack. The proposed method to deduce a CFG from data is described in detail in Sect. 3.

Note that the above CFG creates a necessary but not sufficient description. For example, a million dresser and a bed in a bedroom is a valid configuration by the grammar. Likewise, the relative orientation and shape are not included in the grammar, therefore a scene consisting of couple of small beds on a huge pillow is also a valid scene under the grammar. However, these issues are handled further by the co-occurrence distributions learned by the autoencoder.

2.3 The VAE Network

Let $\mathcal{D}$ be a set of scenes comprising multiple objects. Let $\mathcal{S}_i^j$ be the (bounding box) shape parameters and $\mathcal{P}_i^j = (T_i^j;\ \gamma_i^j)$ be the (absolute) pose parameters of jth object in the ith scene where T_i^j is the center and γ_i^j is the (yaw) angle corresponding to the direction of the object in the horizontal plane, respectively. Note that an object bounding box is aligned with gravity, thus there is only one degree of freedom in its orientation. The world co-ordinates are aligned with the camera co-ordinate frame.

The pose and shape attributes $\Theta^{j\to k} = (\mathcal{P}_i^{j\to k}, \mathcal{S}_i^k)$ are associated with a production rule in which a non-terminal X_j yields a terminal X_k. The pose parameters $\mathcal{P}_i^{j\to k}$ of the terminal object X_k are computed w.r.t. the non-terminal object X_j on the left of the production rule. *i.e.* $\mathcal{P}_i^{j\to k} = (\mathcal{P}_i^j)^{-1}\mathcal{P}_i^k$. The absolute poses of the objects are determined by chaining the relative poses on the path from the root node to the terminal node in the parse tree (see Fig. 1). Note that pose and shape attributes of the production rules corresponding to `None` object are fixed to zero.

The VAE must encode and decode both production rules (1-hot vectors) and the corresponding pose and shape parameters. We achieve this by having separate initial branches of the encoder into which the attributes $\Theta^{j\to k}$, and the 1-hot vectors are passed. Features from the 1-hot encoding branch and the pose-shape branch are then concatenated after a number of 1D convolutional layers. These concatenated features undergo further 1D convolutional layers before being flattened and mapped to the latent space (thereby predicting $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ of $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$). The decoding network is a recurrent network consisting of a stack of GRUs, that takes samples $\boldsymbol{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ (employing reparameterization trick [12]) and outputs logits (corresponding to the production rules) and corresponding attributes $\Theta^{j\to k}$. Logits corresponding to invalid production rules are masked out.

The reconstruction loss of our SG-VAE consists of two parts: **(i)** a cross entropy loss corresponding to the 1-hot encoding of the production rules—note that *soft-max* is computed only on the components after mask-out—and **(ii)** a mean squared error loss corresponding to the production rule attributes (but omitting the terms of `None` objects). Thus, the loss is given as follows:

$$\mathcal{L}_{total}(\phi, \theta; \mathcal{X}, \Theta) = \mathcal{L}_{vae}(\phi, \theta; \mathcal{X}) + \lambda_1 \Big(\mathcal{L}_{pose}(\phi, \theta; \mathcal{P}) + \lambda_2 \mathcal{L}_{shape}(\phi, \theta; \mathcal{S}) \Big) \quad (1)$$

where $\mathcal{L}_{vae}$ is the autoencoder loss [14], and $\mathcal{L}_{pose}$ and $\mathcal{L}_{shape}$ are mean squared error loss corresponding to pose and shape parameters, respectively; ϕ, and θ are the encoder and decoder parameters of the autoencoder that we optimize; $(\mathcal{X}, \Theta)$ are the set of training examples comprising 1-hot encoders and rule attributes. Instead of directly regressing the orientation parameter, the respective *sines* and *cosines* are regressed. Our choice is $\lambda_1 = 10$ and $\lambda_2 = 1$ in all experiments.

3 Discovery of the Scene Grammar

In much previous work a grammar is manually specified. However in this work we aim to discover a suitable grammar for scene layouts in a data-driven manner. It comprises two parts. First we generate a causal graph of all pairwise relationships discovered in the training data, as described in more detail in Sect. 3.1. Second we prune this causal graph by removing all but the dominant discovered relationships, as described in Sect. 3.2.

3.1 Data-Driven Relationship Discovery

We aim to discover causal relationships of different objects that reflects the influence of the appearance (*i.e.* pose and shape) of one object to another. We learn the relationship using hypothesis testing, with each successful hypothesis added to a causal graph (directed) $\mathcal{G} : (\mathcal{V},\ \mathcal{E})$ where the vertex set $\mathcal{V} = \{X_1, \ldots, X_n\}$ is the set of different object categories, and edge set $\mathcal{E}$ is the set of causal relationships. An edge $(X_j \circ\!\!\rightarrow X_{j'}) \in \mathcal{E}$ corresponds to a direct causal influence on occurrence of the object X_j to the object $X_{j'}$. We conduct separate **(i)** appearance based and **(ii)** co-occurrence based testing for causal relationships between a pair of object categories as set out below.

Algorithm 1: χ^2-test for conditional independence check

Input: Co-occurrences O of the objects $X_j, X_{j'}, X_k$

Output: **True** if $X_j \perp\!\!\!\perp X_{j'} \mid X_k$ and **False** Otherwise

1 $$\chi^2 = \sum_{j,j',k \in (\{0,1\})^3} \frac{\left(O_{j,j',k} - \frac{O_{j,k} O_{j',k}}{O_k}\right)^2}{\frac{O_{j,k} O_{j',k}}{O_k}},$$

$O_{j,j',k}$: frequency of occurrences of (j, j', k),
$O_{j,k}$: frequency of occurrences of (j, k),
O_k : frequency of occurrences of k,
N : number of scenes

2 Compute p-value from cumul. χ^2 distrib. with above χ^2 value and d.o.f. ; `/* D.o.f is 2 */`

3 **return** p-value $< \tau$ (we choose $\tau = 0.05$)

(i) Testing for Dependency Based on Co-occurrences. We seek to capture loose associations (*e.g.* sofa and TV) and determine if these associations have a potentially causal nature. To do so, for each pair of object categories we then consider whether these categories are dependent, given a third category. This is performed using the Chi-squared (χ^2) test described below. If the dependence persists across all possible choices of the third category, we conclude that the dependence is not induced by another object, therefore a potentially causal link should exist between them. This exhaustive series of tests is $O(Nm^3)$, where N is the number of scenes and m is the number of categories (in our case, 84). However it is performed offline and only once. This procedure creates an undirected graph with links between pairs where a causal relationship is hypothesized to exist. To establish the direction of causation—*i.e.* turn the undirected graph into a directed one, we use Pearl's Inductive Causation algorithm [20] and the procedure is summarized in Algorithm 2 of the supplementary.

In more detail the χ^2-test checks for conditional independence of a pair of object categories $\{X_j, X_{j'}\}$ given an additional object category $X_k \in \mathcal{V} \setminus \{X_j, X_{j'}\}$. The probabilities required for the test are obtained from the relative

frequencies of the objects and their co-occurrences in the dataset. Algorithm 1 describes this in detail. By way of example, *pillow* and *blanket* might co-occur in a substantial number of scenes, however, their co-occurrences are influenced by a third object category *bed*. In this case, the pairwise relationship between *pillow* and *blanket* is determined to be independent, given the presence of *bed*, so no link between *pillow* and *blanket* is created.

(ii) Testing for Dependency Based on Shape and Pose. In addition to the conditional co-occurrence captured above, we also seek to capture covering/enclosing and supporting relationships—which are defined by the shape and pose of the objects as well as the categories—in the causal graph. More precisely we hypothesize a causal relationship between object categories A and B if:

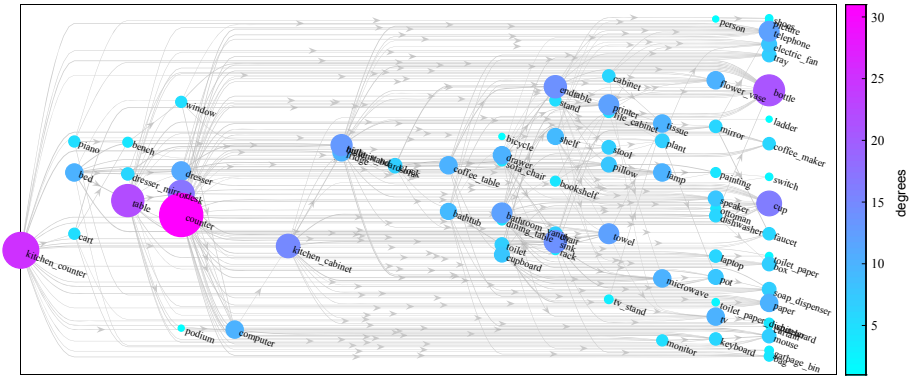


Fig. 2. The above graph is generated by the modified IC algorithm on SUNRGBD-3D Dataset. An arrowhead of an edge indicates the direction of causation and the color of a node indicates its degree.

- object category A is found to support object category B (*i.e.* their relative poses and shapes are such that, within a threshold, one is above and touching the other), or,
- object category B is enclosed/covered by another larger object category A (again this is determined using a threshold on the objects' relative shapes and poses).

We accept a hypothesis and establish the causal relationship (by entering a suitable edge into the causal graph) if at least 30% of the co-occurrences of these object categories in the dataset agree with the hypothesis. The final directed causal relational graph $\mathcal{G}$ is the union of the causal graphs generated by the above tests. Note that we do not consider any dependencies that would lead to a cycle in the graph [4]. The result of the procedure is also displayed in Fig. 2.

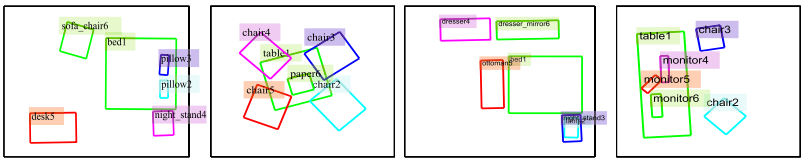
3.2 Creating a CFG from the Causal Graph

We now need to create a Context Free Grammar from the causal graph. The CFG is characterised by non-terminal symbols that generate other symbols. Suppose we choose a particular node in the causal graph (*i.e.* an object category) and assume it is non-terminal. By tracing the full set of directed edges in the causal graph emanating from this node we create a set of production rules. This non-terminal and associated rules are then tested against the dataset to determine how many scenes are explained (formally "covered") by the rules. A good choice of non-terminals will lead to good coverage. Our task then is to determine an optimal set of non-terminals and associated rules to give the best coverage of the full dataset of scenes.

Note that finding such a set is a combinatorial hard problem. Therefore, we devise a greedy algorithm to select non-terminals and find approximate best coverage. Let X_j, an object category, be a potential non-terminal symbol and $\mathcal{R}_j$ be the set of production rules derived from X_j in the causal graph. Let C_j be the set of terminals that $\mathcal{R}_j$ covers (essentially nodes that X_j leads to in $\mathcal{G}$). Our greedy algorithm begins with an empty set $\mathcal{R} = \emptyset$ and chooses the node X_j and associated production rule set $\mathcal{R}_j$ to add that maximize the *gain* in coverage $\mathcal{G}_{gain}(\mathcal{R}_j, \mathcal{R}) = \frac{1}{|\mathcal{R}_j|} \sum_{I_i \in \mathcal{I} \setminus \mathcal{C}} |Y_i|/|I_i|$.

Unique Parsing. Given a scene there could be multiple parse trees derived by the leftmost derivation grammar and hence produces different sequences and different representations. For example, for a scene consist of `bed`, `sofa` and `pillow`, the terminal object `pillow` could be generated by any of the non-terminals `BED` or `SOFA`. This ambiguity can confuse the parser while encoding a scene. Further, different orderings of multiple occurrences of an object lead to different parse trees. We consider following parsing rules to remove the ambiguity:

- Fix the order of the object categories in the order of precedence defined by the grammar.
- Multiple occurrences of an object are sorted in the appearance w.r.t. the preceded object in the anti-clockwise direction starting from the object making minimum angle to the orientation of the preceded object. One such example is shown in Fig. 3. Note that this ordering is only required during training.



(a) Ground-truth 1 (b) Ground-truth 2 (c) Ground-truth 3 (d) Ground-truth 4

Fig. 3. The parsing order is displayed by a numeral concatenated with the object name. (a) the objects are sorted in the order of precedence defined by the grammar, and (b) multiple chairs are sorted in the appearance w.r.t. the table in anti-clockwise direction starting from the bottom right corner. (c)–(d) More examples of the object order in the ground-truth samples from SUN RGB-D dataset [26].

4 Experiments

Dataset. We evaluate the proposed method on SUN RGB-D Dataset [26] consisting of 10,335 real scenes with 64,595 3D bounding boxes and about 800 object categories. The dataset is a collection of multiple datasets [10,25,32], and is highly unbalanced: *e.g.* a single object category *chair* corresponds to about 31% of all the bounding boxes and 38% of total object categories occur just once in the entire dataset. We consider object categories appearing at least 10 times in the dataset for evaluation. Further, very similar object categories are merged, yielding 84 object categories and 62,485 bounding boxes.

Table 1. Results of 3D bounding box reconstruction of some of the frequent objects under a valid reconstruction with IoU >0.25 [21]. The pose estimation results are furnished within braces (angular errors in degrees, displacement errors in meters).

Objects	chair	bed	table	ktchn_cntr	piano
SG-VAE	**92.3**(**5.88°**,**0.13m**)	**100.0**(**2.80°**,**0.08 m**)	**96.2**(**2.84°**,**0.08 m**)	**100.0**(**1.69°**,**0.08 m**)	**67.1**(**6.93°**,**0.11 m**)
BL1	75.4 (8.30°, **0.14 m**)	98.5 (5.70°, **0.09 m**)	93.8 (3.70°, **0.08 m**)	**100.0**(**1.62°**,**0.06 m**)	48.7 (10.3°, **0.12 m**)
BL2 [6]	33.7 (28.1°, 0.32 m)	75.1 (7.12°, 0.45 m)	90.2 (7.69°, 0.33 m)	83.2 (5.96°, 0.09 m)	0.80 (45.1°, 0.43 m)
BL3[14]+[34]	29.2 (41.8°, 0.26 m)	88.7 (35.7°, 0.58 m)	72.9 (56.7°, 0.34 m)	100.0 (52.5°, 0.78 m)	26.3 (17.4°, 0.33 m)

We separated 10% of the data at random for validation. On average there are 4.29 objects per image and maximum number of objects in an image is considered to be 15. This also provides the upper bound of the length of the sequence generated by the grammar. Note that the dataset is the intersection of the given dataset and the scene language, *i.e.* the possible set of scenes generated by the CFG.

Table 2. IoU of room layout estimation

Methods	SG-VAE	BL1	BL2 [6]	BL3 [14]+[34]
Grammar	✓	✓		✓
Pose & Shape	✓	✓	✓	
IoU	**0.6240**	0.5673	0.2964	0.5119

Baseline Methods for Evaluation (Ablation Studies). To evaluate the individual effects of (i) output of the decoder structure, (ii) usage of grammar, and (iii) the pose and shape attributes, the following baselines are chosen:

(**BL**1) *Variant of SG-VAE*: In contrast to the proposed SG-VAE where attributes of each rule are directly concatenated with 1-hot encoding of the rule, in this variant separate attributes for each rule type are predicted by the decoder and rest are filled with zeros.

(**BL**2) *No Grammar VAE* [6]: No grammar is considered in this baseline. The 1-hot encodings correspond to the object type is concatenated with the absolute pose of the objects (in contrast to rule-type and relative pose in SG-VAE) respectively.

(**BL**3) *Grammar VAE* [14] + *Make home* [34]: The Grammar VAE is incorporated with our extracted grammar to sample a set of coherent objects and [34] is used to arrange them. Sampled 10 times and solution corresponding to best IoU w.r.t. groundtruth is employed. The details of the above baselines are provided in the supplementary.

All the baselines including SG-VAE are implemented in `python 2.7 (Tensorflow)` and trained on a GTX 1080 Ti GPU.

Evaluation Metrics. The relative poses of individual objects in a scene are accumulated to compute their absolute poses which are then combined with the shape parameters to compute the scene layout. The reconstructed scene layouts are then compared against the groundtruth layouts.

- **3D bounding box reconstruction.** We employed IoU to measure the shape similarity of the bounding boxes. Reconstructed bounding boxes with IoU > 0.25 are considered as true positives. The results are reported in Table 1.
- **3D Pose estimation.** The average pose error is considered over two separate metrices: **(i)** angular error in degrees, **(ii)** displacement error in meters (see Table 1).
- **Room Layout Estimation.** The evaluation is conducted as the IoU of the *occupied* space between the groundtruth and the predicted layouts (see Table 2). The intersection is computed only over the true positives.

Interpolation in Latent Space. Two distinct scenes are encoded into the latent space, *e.g.*, $\mathcal{N}(\boldsymbol{\mu_1}, \boldsymbol{\Sigma_1})$ and $\mathcal{N}(\boldsymbol{\mu_2}, \boldsymbol{\Sigma_2})$ and new scenes are then synthesized from interpolated vectors of the means, *i.e.* from $\alpha\boldsymbol{\mu_1} + (1-\alpha)\boldsymbol{\mu_2}$. We performed the experiment on a set of random pairs chosen from the test dataset. The results are shown in Fig. 4. Notice that the decoder behaves gracefully w.r.t. perturbations of the latent code and always yields a valid and realistic scene.

4.1 Comparison with Baselines on Other Datasets

The conventional indoor scene synthesis methods (for example, Grains [16], Human-centric [23] (HC), fast-synth [24] (FS) etc.) are tailored to and trained on SUNCG dataset [28]. The dataset consists of synthetic scenes generated by graphic designers. Moreover, the dataset is no longer publicly available (along with the meta-files).[1] Therefore the following evaluation protocols are employed to assess the performance of different methods.

Comparison on Synthetic Scene Quality. To conduct a qualitative evaluation, we employ a classifier (based on Pointnet [22]) to predict a scene layout to

[1] Due to the legal dispute around SUNCG [28] we include our results for SUNCG (conducted on our internal copy) in Table 3 only for illustrative purposes.

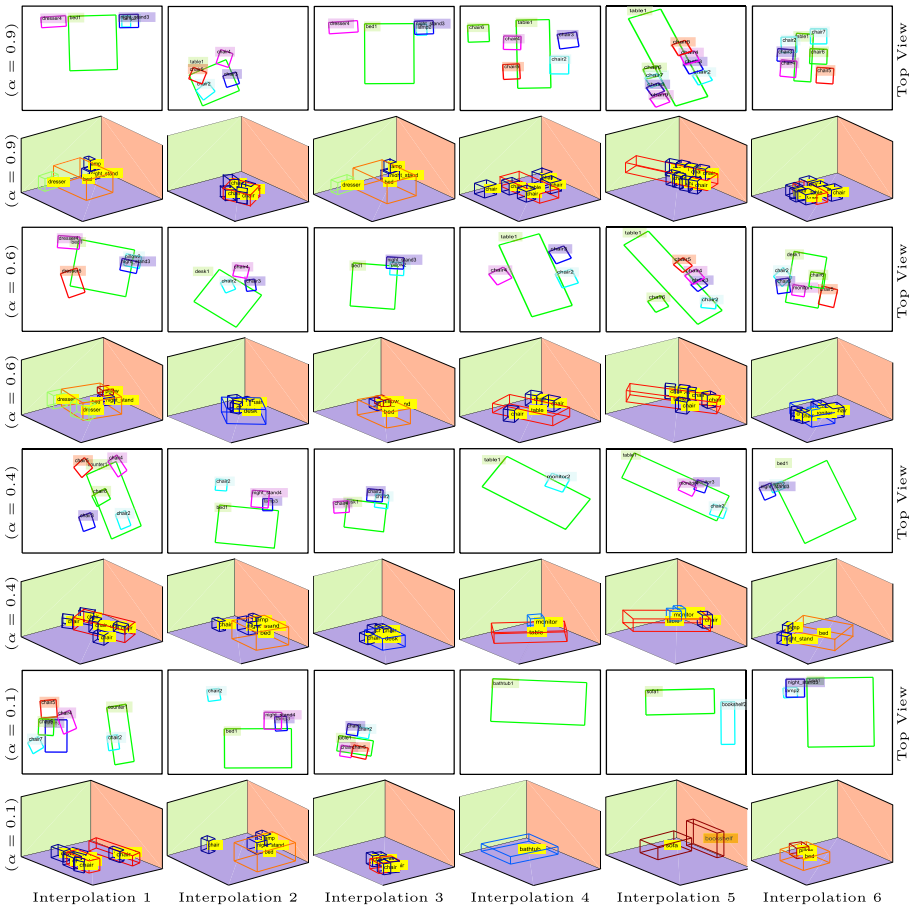


Fig. 4. Synthetic scenes decoded from linear interpolations $\alpha\boldsymbol{\mu_1} + (1-\alpha)\boldsymbol{\mu_2}$ of the means $\boldsymbol{\mu_1}$ and $\boldsymbol{\mu_2}$ of the latent distributions of two separate scenes. The generated scenes are valid in terms of the co-occurrences of the object categories and their shapes and poses (more examples can be found in the supplementary). The room-size and the camera view-point are fixed for better visualization. Best viewed electronically.

be an original or generated by a scene synthesis method. If the generated scenes are very similar to the original scenes, the classifier performs poorly (lower accuracy) and indicates the efficacy of the synthesis method. The classifier takes a scene layout of multiple objects, individually represented by the concatenation of 1-hot code and the attributes, as input and predicts a binary label according to the scene-type. The classifier is trained and tested on a dataset of $2K$ original and synthetic scenes (50% training and 50% testing). Note that SG-VAE is trained on the synthetic data generated by the interpolations of latent vectors (some examples are shown in Fig. 4) and real data of SUN RGB-D [26]. Lower accuracy of the classifier validates the superior performance of the proposed

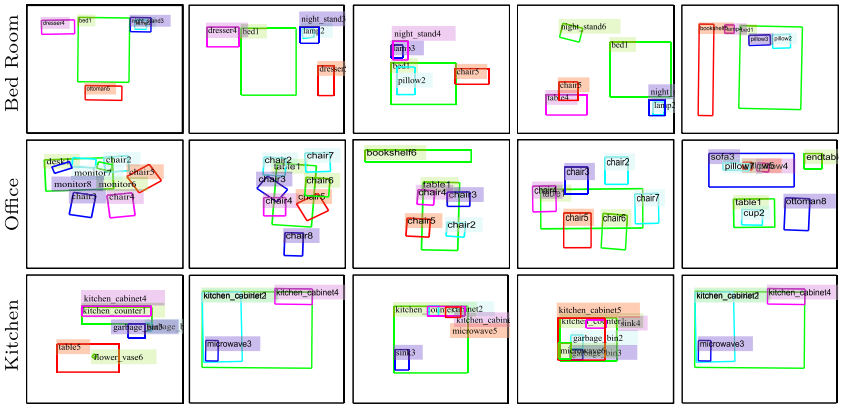


Fig. 5. Top-views of the synthesized scenes generated by the SG-VAE on SUN RGB-D. A detailed comparison with other baselines on SUNCG can be found in supplementary.

SG-VAE. The average performance is plotted in Table 3. Examples of some synthetic scenes[2] generated by SG-VAE are also shown in Fig. 5.

Table 3. Original vs. synthetic classification accuracy (trained with Pointnet [22]): The accuracy indicates that the synthetic scenes are indistinguishable from the original scenes and hence lower (closer to 50%) is better.

Datasets	SUN RGB-D [26]	SUNCG [28]		
Methods	SG-VAE	SG-VAE	Grains [16]	HC [23]
Accuracy	**71.3**%	**83.7**%	96.4%	98.1%

Runtime Comparison. All the methods are evaluated on a single CPU and the runtime is displayed in Table 4. Note that the decoder of the proposed SG-VAE takes only ∼1 ms to generate the parse tree and the rest of the time is consumed by the renderer (generating bounding boxes). The proposed method is almost two orders of magnitude faster than the other scene synthesis methods.

Table 4. Average time required to generate a single scene.

Methods	SG-VAE	Grains [16]	FS [24]	HC [23]
Avg. runtime	**8.5 ms**	1.2×10^2 ms	1.8×10^3 ms	2.4×10^5 ms

[2] We thank the authors of Grains [16] and HC [23] for sharing the code. More results and the proposed SG-VAE for SUNCG are in the supplementary material.

4.2 Scene Layout Estimation from the RGB-D Image

The task is to predict the 3D scene layout given an RGB-D image. Typically, the state of the art methods are based on sophisticated region proposals and subsequent processing [21,27]. With this experiment, we aim to demonstrate the potential use of the latent representation learned by the proposed auto-encoder for a computer vision task, and therefore we employ a simple approach at this point. We (linearly) map deep features (extracted from images by a DNN [38]) to the latent space of the scene-grammar autoencoder. The decoder subsequently generates a 3D scene configuration with associated bounding boxes and object labels from the projected latent vector. Since during the deep feature extraction and the linear projection, the spatial information of the bounding boxes are lost, the predicted scene layout is then combined with a bounding box detection to produce the final output.

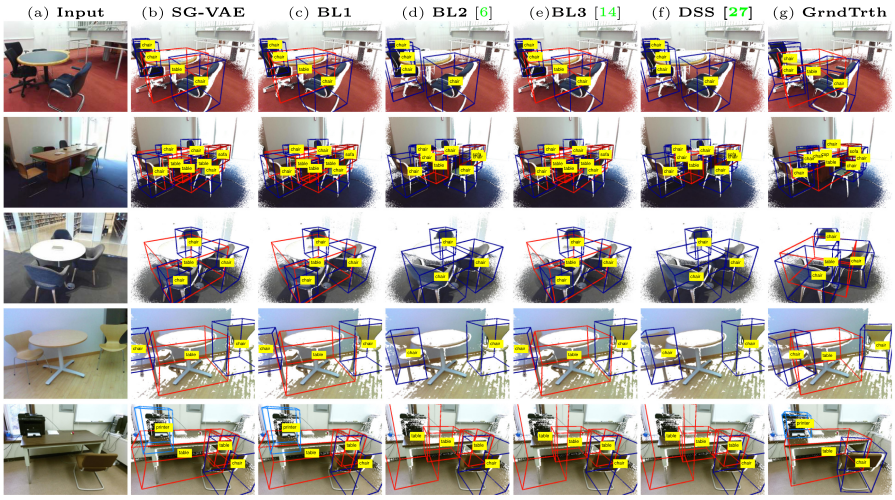


Fig. 6. A few results on SUNRGB Dataset inferred from the RGB-D images.

The bounding box detector of DSS [27] is employed and the scores of the detection are updated based on our reconstruction as follows: the score (confidence of the prediction) of a detected bounding box is doubled if a similar bounding box (in terms of shape and pose) of the same category is reconstructed by our method. A 3D non-maximum suppression is applied to the modified scores to get the final scene layout. The details can be found in the supplementary.

We selected the average IoU for room layout estimation as the evaluation metric, and the results are presented in Table 5. The proposed method and other grammar-based baselines improve the scene layout estimation from the same by sophisticated methods such as deep sliding shapes [27]. Furthermore, the proposed method tackles the problem in a much simpler and faster way. Thus, it can be

Table 5. IoU for RGBD to room layout estimation

Methods	SG-VAE	BL1	BL2 [6]	BL3 [14]+[34]	DSS [27]
IoU	**0.4387**	0.4315	0.4056	0.4259	0.4070

employed to any 3D scene layout estimation method with very little overhead (*e.g.* a few ms in addition to 5.6 s of [27]). Results on some test images where SG-VAE produces better IoUs are displayed in Fig. 6.

5 Related Works

The most relevant method to ours is Grains [16]. It requires training separate networks for each of the room-types—bedroom, office, kitchen etc. HC [23] is very slow and takes a few minutes to synthesize a single layout. FS [24] is fast, but still takes a couple of seconds. SceneGraphNet [33] predicts a probability distribution over object types that fits well in a target location given an incomplete layout. A similar graph-based method is proposed in [29] and CNN-based method proposed in [30]. A complete survey of the relevant method can be found in [35]. Note that all these methods are tailored to and trained on the synthetic SUNCG dataset which is currently unavailable.

Koppula *et al.* [13] propose a graphical model that captures the local visual appearance and co-occurrences of different objects in the scene. They learn the appearance relationships among objects from the visual features that takes an RGB-D image as input and predicts 3D semantic labels of the objects as output. The pair-wise support relationships of the indoor objects are also exploited in [8,25]. Learning to 3D scene synthesis from annotated RGB-D images is proposed in [11]. In the similar direction, an example-based synthesis of 3D object arrangements is proposed in [5].

Grammar-based models for 3D scene reconstruction have been partially exploited before [36,37], *e.g.* textured probabilistic grammar [15]. Zhao *et al.* [36] proposed handcoded grammar to its terminal symbols (line segments) and later extended to different functional groups in [37]. Choi *et al.* [2] proposed a 3D geometric phrase model that estimates a scene layout with multiple object interactions. Note that all the above methods are based on hand-coded production rules, in contrast, the proposed method exploits a self-supervision to yield the production rules of the grammar.

6 Conclusion

We proposed a grammar-based autoencoder SG-VAE for generating natural indoor scene layouts containing multiple objects. By construction the output of SG-VAE always yields a valid configuration (w.r.t. the grammar) of objects, which was also experimentally confirmed. We demonstrated that the obtained

latent representation of an SG-VAE has desirable properties such as the ability to interpolate between latent states in a meaningful way. The latent space of SG-VAE can also be easily adapted to computer vision problems (*e.g.* 3D scene layout estimation from RGB-D images). Nevertheless, we believe that there is potential in leveraging the latent space of SG-VAEs to the other tasks, *e.g.* fine-tuning the latent space for a consistent layout over multiple cameras which is part of the future work.

Acknowledgement. We gratefully acknowledge the support of the Australian Research Council through the Centre of Excellence for Robotic Vision, CE140100016, and the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

References

1. Anderson, P., et al.: Bottom-up and top-down attention for image captioning and visual question answering. In: Proceedings of CVPR, pp. 6077–6086 (2018)
2. Choi, W., Chao, Y.W., Pantofaru, C., Savarese, S.: Understanding indoor scenes using 3D geometric phrases. In: Proceedings of CVPR, pp. 33–40 (2013)
3. Deng, C., Wu, Q., Wu, Q., Hu, F., Lyu, F., Tan, M.: Visual grounding via accumulated attention. In: Proceedings of CVPR, pp. 7746–7755 (2018)
4. Dor, D., Tarsi, M.: A simple algorithm to construct a consistent extension of a partially oriented graph (1992)
5. Fisher, M., Ritchie, D., Savva, M., Funkhouser, T., Hanrahan, P.: Example-based synthesis of 3D object arrangements. ACM Trans. Graph. (TOG) **31**(6), 1–11 (2012)
6. Gómez-Bombarelli, R., et al.: Automatic chemical design using a data-driven continuous representation of molecules. ACS Cent. Sci. **4**(2), 268–276 (2018)
7. Goodfellow, I., et al.: Generative adversarial nets. In: Proceedings of NIPS, pp. 2672–2680 (2014)
8. Guo, R., Hoiem, D.: Support surface prediction in indoor scenes. In: Proceedings of ICCV, pp. 2144–2151 (2013)
9. Huang, S., Qi, S., Zhu, Y., Xiao, Y., Xu, Y., Zhu, S.-C.: Holistic 3D scene parsing and reconstruction from a single RGB image. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) ECCV 2018. LNCS, vol. 11211, pp. 194–211. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01234-2_12
10. Janoch, A., et al.: A category-level 3D object dataset: putting the kinect to work. In: Fossati, A., Gall, J., Grabner, H., Ren, X., Konolige, K. (eds.) Consumer Depth Cameras for Computer Vision. ACVPR, pp. 141–165. Springer, London (2013). https://doi.org/10.1007/978-1-4471-4640-7_8
11. Kermani, Z.S., Liao, Z., Tan, P., Zhang, H.: Learning 3D scene synthesis from annotated RGB-D images. Comput. Graph. Forum **35**, 197–206 (2016). Wiley Online Library
12. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. In: Proceedings of ICLR, pp. 469–477 (2014)
13. Koppula, H.S., Anand, A., Joachims, T., Saxena, A.: Semantic labeling of 3D point clouds for indoor scenes. In: Proceedings of NIPS, pp. 244–252 (2011)
14. Kusner, M.J., Paige, B., Hernández-Lobato, J.M.: Grammar variational autoencoder. In: Proceedings of ICML, pp. 1945–1954. JMLR.org (2017)

15. Li, D., Hu, D., Sun, Y., Hu, Y.: 3D scene reconstruction using a texture probabilistic grammar. Multimedia Tools Appl. **77**(21), 28417–28440 (2018)
16. Li, M., et al.: GRAINS: generative recursive autoencoders for indoor scenes. ACM Trans. Graph. (TOG) **38**(2), 12 (2019)
17. Liu, M.Y., Tuzel, O.: Coupled generative adversarial networks. In: Proceedings of NIPS, pp. 469–477 (2016)
18. Lu, J., Yang, J., Batra, D., Parikh, D.: Hierarchical question-image co-attention for visual question answering. In: Proceedings of NIPS, pp. 289–297 (2016)
19. Meyer, J.A., Filliat, D.: Map-based navigation in mobile robots: II. A review of map-learning and path-planning strategies. Cogn. Syst. Res. **4**(4), 283–317 (2003)
20. Pearl, J., Verma, T.S.: A theory of inferred causation. In: Studies in Logic and the Foundations of Mathematics, vol. 134, pp. 789–811. Elsevier (1995)
21. Qi, C.R., Chen, X., Litany, O., Guibas, L.J.: ImVoteNet: boosting 3D object detection in point clouds with image votes. In: Proceedings of CVPR, pp. 4404–4413 (2020)
22. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: PointNet: deep learning on point sets for 3D classification and segmentation. In: Proceedings of CVPR, pp. 652–660 (2017)
23. Qi, S., Zhu, Y., Huang, S., Jiang, C., Zhu, S.C.: Human-centric indoor scene synthesis using stochastic grammar. In: Proceedings of CVPR, pp. 5899–5908 (2018)
24. Ritchie, D., Wang, K., Lin, Y.: Fast and flexible indoor scene synthesis via deep convolutional generative models. In: Proceedings of CVPR, pp. 6182–6190 (2019)
25. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from RGBD images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012. LNCS, vol. 7576. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33715-4_54
26. Song, S., Lichtenberg, S.P., Xiao, J.: Sun RGB-D: a RGB-D scene understanding benchmark suite. In: Proceedings of CVPR, pp. 567–576 (2015)
27. Song, S., Xiao, J.: Deep sliding shapes for amodal 3D object detection in RGB-D images. In: Proceedings of CVPR, pp. 808–816 (2016)
28. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. In: Proceedings of CVPR, pp. 1746–1754 (2017)
29. Wang, K., Lin, Y.A., Weissmann, B., Savva, M., Chang, A.X., Ritchie, D.: PlanIT: planning and instantiating indoor scenes with relation graph and spatial prior networks. ACM Trans. Graph. (TOG) **38**(4), 1–15 (2019)
30. Wang, K., Savva, M., Chang, A.X., Ritchie, D.: Deep convolutional priors for indoor scene synthesis. ACM Trans. Graph. (TOG) **37**(4), 1–14 (2018)
31. Xiao, F., Sigal, L., Jae Lee, Y.: Weakly-supervised visual grounding of phrases with linguistic structures. In: Proceedings of CVPR, pp. 5945–5954 (2017)
32. Xiao, J., Owens, A., Torralba, A.: SUN3D: a database of big spaces reconstructed using SfM and object labels. In: Proceedings of ICCV, pp. 1625–1632 (2013)
33. Zhou, Y., While, Z., Kalogerakis, E.: SceneGraphNet: neural message passing for 3D indoor scene augmentation. In: Proceedings of ICCV (2019)
34. Yu, L.F., Yeung, S.K., Tang, C.K., Terzopoulos, D., Chan, T.F., Osher, S.J.: Make it home: automatic optimization of furniture arrangement. ACM Trans. Graph. (TOG) **30**, 86 (2011)
35. Zhang, S.H., Zhang, S.K., Liang, Y., Hall, P.: A survey of 3D indoor scene synthesis. J. Comput. Sci. Technol. **34**(3), 594–608 (2019)
36. Zhao, Y., Zhu, S.C.: Image parsing with stochastic scene grammar. In: Proceedings of NIPS, pp. 73–81 (2011)

37. Zhao, Y., Zhu, S.C.: Scene parsing by integrating function, geometry and appearance models. In: Proceedings of CVPR, pp. 3119–3126 (2013)
38. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. In: Proceedings of NIPS, pp. 487–495 (2014)