CGI2019 manuscript No.
(will be inserted by the editor)

Stylistic Scene Enhancement GAN: Mixed Stylistic Enhancement Generation for 3D Indoor Scenes

Suiyun Zhang · Zhizhong Han · Yu-Kun Lai · Matthias Zwicker · Hui Zhang*

Abstract In this paper, we present stylistic scene enhancement GAN, SSE-GAN, a conditional Wasserstein GAN-based approach to automatic generation of mixed stylistic enhancements for 3D indoor scenes. An enhancement indicates factors that can influence the style of an indoor scene such as furniture colors and occurrence of small objects. To facilitate network training, we propose a novel enhancement feature encoding method, which represents an enhancement by a multi-one-hot vector, and effectively accommodates different enhancement factors. A Gumbel-Softmax module is introduced in the generator network to enable the generation of high fidelity enhancement features that can better confuse the discriminator. Experiments show that our approach is superior to the other baseline methods and successfully models the relationship between the style distribution and scene enhancements. Thus, although only trained with a dataset of room images in single styles, the trained generator can generate mixed stylistic enhancements by specifying multiple styles as the condition. Our approach is the first to apply a Gumbel-Softmax module in conditional Wasserstein GANs, as well as the first to explore the application of GAN-based models in the scene enhancement field.

Suiyun Zhang · Hui Zhang (* Corresponding author)
[1] School of Software, Tsinghua University, Beijing, China
[2] Beijing National Research Center for Information Science and Technology (BNRist), Beijing, China
E-mail: {zhangsuiyun13@mails., huizhang@}tsinghua.edu.cn

Zhizhong Han · Matthias Zwicker
Department of Computer Science, University of Maryland, Maryland, College Park, MD 20742, United States
E-mail: h312h@umd.edu, zwicker@cs.umd.edu

Yu-Kun Lai
School of Computer Science & Informatics, Cardiff University, Wales, United Kingdom.
E-mail: Yukun.Lai@cs.cardiff.ac.uk

Keywords Scene enhancement · 3D indoor scenes · Interior design · Conditional generative adversarial nets · Gumbel-Softmax · Multi-one-hot

1 Introduction

It is very common that an indoor room possesses more than one interior style (e.g., both *modern* and *girl*). Automatic generation of mixed stylistic enhancements for a 3D indoor scene is crucial for constructing realistic virtual environments in a plethora of applications such as computer games, 3D movies, and interior design. An enhancement for an indoor scene in this paper refers to the aspects that can influence the style of the scene, such as furniture colors, geometry, and small object occurrence. Although it is trivial to obtain 3D indoor scenes from online 3D repositories [1], 3D indoor scene dataset [27], and automatic scene synthesis methods [29], these scenes usually cannot be characterized by one or more specific styles such as *romantic*, *modern*, etc. Moreover, it is time-consuming for designers to manually create mixed stylistic enhancements, which involves carefully selecting the color for each furniture object and picking specific small objects to place into the 3D scene while regarding the semantics of the styles. Therefore, an approach that can automatically generate enhancements of one or more styles is highly demanded to save tedious labor work.

A few recent works have been proposed to tackle the problem of stylistic enhancement generation for 3D indoor scenes. However, they either only consider one aspect of enhancement factors at a time, e.g., changing furniture materials [6,8], or can only produce scene enhancements of a single interior style such as *romantic* [6,33]. Thus, these methods cannot generate mixed

stylistic enhancements for 3D indoor scenes while taking into account multiple enhancement factors.

To address this challenge, we need to model the underlying distribution of 3D indoor scene enhancements conditioned on a few style words, which might be multimodal and quite complex. Inspired by the impressive success in applying conditional Generative Adversarial Networks (cGAN) [23] to many conditional generation tasks such as synthesizing images from text descriptions [26], we base our approach on cGAN. However, it is tricky to uniformly represent different enhancement factors such as furniture colors and small object occurrence in a form that can be fed into the networks. Thus, a dedicated encoding scheme is required.

In this paper, we propose stylistic scene enhancement GAN (SSE-GAN), a novel GAN-based approach to stylistic scene enhancement generation. We use two enhancement factors introduced in [33] for illustrating our method, including color for furniture categories and occurrence for small object categories. Specifically, an image dataset with style labels is used as our training data. To enable the learning of SSE-GAN, we extract the enhancement information in each image and represent it with an enhancement vector through a novel encoding method. SSE-GAN regards a style as the condition and generates enhancement feature vectors that are consistent with the style. The generator and discriminator in SSE-GAN are trained adversarially. To generate enhancement feature vectors that are hard to be distinguished from the real ones by the discriminator, we introduce a novel Gumbel-Softmax module in the generator. We also utilize Wasserstein GAN (W-GAN) architecture [13] to stabilize the training process and increase the diversity of generated enhancements.

SSE-GAN learns the enhancement distribution conditioned on different styles. Thus, it can generate mixed stylistic scene enhancements even though it is trained by scenes of single styles. Therefore, with the trained generator, diverse enhancement feature vectors representing one or more scene styles can be generated by manipulating the condition styles. These feature vectors can then be applied to 3D indoor scenes to obtain the final enhancement results. To the best of our knowledge, we are the first to introduce the conditional Wasserstein GAN with a Gumbel-Softmax module and to address the scene enhancement problem with a GAN-based approach.

Our main contributions are as follows:

- We propose SSE-GAN to generate stylistic scene enhancements of one or more styles for 3D indoor scenes. The trained generator can generate single, mixed and in-between stylistic enhancements with different styles.
- We introduce a novel scene enhancement encoding method to accommodate different enhancement factors such as furniture colors and small object occurrence into a feature vector, which enables the learning and generation of scene enhancements by deep neural networks.
- We introduce a novel Gumbel-Softmax module to the generator architecture that facilitates the generation of high fidelity enhancement feature vectors.

2 Related Work

2.1 Stylistic scene enhancement

In this paper, an enhancement of a 3D scene includes coloring furniture objects and placing small objects into the scene. Therefore, we review the related works about the two aspects accordingly. Zhang et al. [33] proposed a framework that enhances an input 3D scene with a style word by coloring the furniture objects and placing additional small objects. Chen et al. [6] trained a Bayesian network to recommend colors for furniture objects concerning a few styles. However, both methods can only take one style word at a time so they cannot be used to generate mixed stylistic enhancements. Moreover, lacking small objects in the results of [6] reduces the realism of the generated enhancements. Image2Scene [9] transfers the style from an image to a 3D indoor scene by recoloring and rearranging the furniture objects. The style can only be captured by one image, and they require the image and the 3D scene be well corresponded. Thus, it is not scalable or flexible to apply their method to other scenes or mixed styles.

Different from these methods, our approach learns the distribution of scene enhancements conditioned on a few style words with a conditional GAN-based model. Thus, mixed stylistic enhancements can be generated by using multiple styles as the condition.

2.2 Generative Adversarial Networks in 3D

Generative adversarial networks (GANs) [12] have been popular since proposed, and have achieved impressive performance in various tasks such as DC-GAN [25] for image generation, CartoonGAN [10] for image stylization, and GP-GAN [30] for high-resolution image blending, etc. Comparatively, due to the increase of dimensionality of 3D, applications of GANs in the 3D field are less. Wu et al. [31] presented 3D-GAN which employs volumetric convolution in GAN to generate 3D voxels. Liu et al. [18] used a GAN to assist users in modeling 3D shapes interactively by creating a projection operator on top of 3D-GAN, which maps a 3D voxel input

to a latent space. Chen et al. [7] proposed a conditional Wasserstein GAN framework [2,13] based method to generate colored 3D shapes from natural language.

While these methods focus on modeling images or 3D shapes, our approach aims at modeling the style of 3D scene enhancements. Our approach is the first to model scene enhancements in a generative adversarial manner. Moreover, although both our method and [7] are based on conditional Wasserstein GAN (cWGAN), a novel Gumbel-Softmax module is introduced in our network architecture to enable the generation of our proposed enhancement feature representation.

2.3 Categorical variable generation

In this paper, we use a one-hot vector to encode the color for a furniture category or the occurrence for a small object category. The one-hot vectors of all the categories are combined to form a multi-one-hot vector to represent an enhancement feature. Each one-hot vector is regarded as a categorical variable while the multi-one-hot enhancement feature vector is regarded as a multi-categorical variable.

However, GANs are mostly applied to generate data in continuous domains, such as images [15], audios [11] and 3D volumes [31]. It is hard to generate categorical data using GANs because it usually involves a non-differentiable sampling process, e.g., an argmax operation after a softmax layer, which makes it hard to calculate the gradients and blocks back-propagation. Two concurrent works tackled the same problem in a very similar way using variational autoencoders (VAE). Specifically, Jang et al. [16] proposed the Gumbel-Softmax distribution and Maddison et al. [20] proposed the Concrete-Distribution. Both distributions are differentiable and can be used to approximate a categorical distribution. Thus, they can be used to replace the previous ill-posed argmax structure in the generative models. As these two distributions are almost the same in their formulations, we refer to them as Gumbel-Softmax in the rest of this paper.

Recently, in order to generate multi-categorical data, Camino et al. [4] applied a Gumbel-Softmax layer in the output layer of several different network structures such as the vanilla GANs [12] and adversarially regularized autoencoders (ARAE) [21]. Different from their work, we introduce the Gumbel-Softmax module in the Wasserstein GAN rather than the vanilla GANs. Moreover, our networks are conditional, which take one or more style words as the condition. As far as we know, we are the first to explore the generation of multi-categorical variables in a conditional manner with a cWGAN-based architecture by a particular Gumbel-Softmax module.

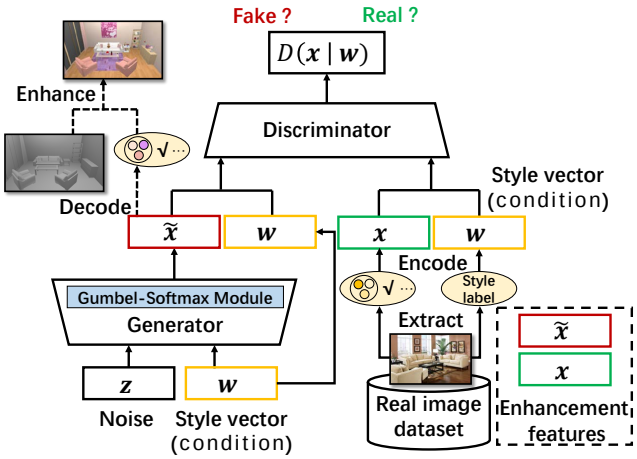


Fig. 1 Overview of our scene enhancement pipeline.

3 Overview

Given an input 3D scene with initially arranged furniture objects and one or more style words, our goal is to suggest a few scene enhancements for the input scene according to the input style word(s). Although there are many factors that may affect the style of a 3D scene, in this paper, we limit our discussion to two factors, including the color of furniture objects and occurrence of particular small object categories.

Fig. 1 summarizes the pipeline of our approach. A labeled indoor room image dataset is used to learn the distribution of scene enhancements conditioned on the style words. The information of furniture colors and occurrence of small objects in each image is extracted and encoded into a scene enhancement vector $\boldsymbol{x}$ by our novel enhancement encoding method. The style label of the image is represented by a one-hot style vector $\boldsymbol{w}$. We will detail the image dataset and the enhancement feature encoding method in Secs. 4.1 and 4.2, respectively.

In order to increase the diversity of the results, we base our generative model on Wasserstein GAN (WGAN) with gradient penalty in the conditional sense. Specifically, the discriminator $\mathcal{D}$ takes a pair of vectors as input, i.e., an enhancement feature vector $\boldsymbol{x}$ and a style vector condition $\boldsymbol{w}$, and outputs a score, $\mathcal{D}(\boldsymbol{x}|\boldsymbol{w})$, which indicates how well $\boldsymbol{x}$ is conditioned on $\boldsymbol{w}$. The generator $\mathcal{G}$ takes a noise vector $\boldsymbol{z}$ and $\boldsymbol{w}$ as input, and generates a 'fake' enhancement feature vector $\tilde{\boldsymbol{x}} = \mathcal{G}(\boldsymbol{z}|\boldsymbol{w})$. Generally, $\mathcal{D}$ tries to distinguish between samples from the real image dataset and samples generated from $\mathcal{G}$, while $\mathcal{G}$ is tasked with generating as real samples as possible to confuse $\mathcal{D}$. The real ($\boldsymbol{x}$) and fake ($\tilde{\boldsymbol{x}}$) samples are highlighted by green and red boxes respectively in Fig. 1. In order to increase the similarity between the real features $\boldsymbol{x}$ and the generated fake features $\tilde{\boldsymbol{x}}$, a special activation function Gumbel-Softmax

is introduced into the generator network to generate high fidelity feature vectors $\tilde{\boldsymbol{x}}$. The architecture and training of $\mathcal{D}$ and $\mathcal{G}$ are detailed in Sec. 5.

As shown by the dashed lines in Fig. 1, with the trained generator and an input 3D scene (the gray image), one can obtain a scene enhancement (the upper left color image) as follows. First, pass a random noise $\boldsymbol{z}$ and a style vector $\boldsymbol{w}$ as the *condition* to the generator. Then, apply the enhancement information decoded from the generated enhancement feature vector $\tilde{\boldsymbol{x}}$ to the input 3D scene. We will detail the feature decoding and enhancement generation in Sec. 4.4.

4 Scene enhancement encoding and generation

4.1 Dataset

As stylistic 3D indoor scenes are very limited, we use the image dataset from [33] as the training data. The dataset contains 400 images in total, and each image is labeled with a style word. Let $\boldsymbol{Y} = \{y_k \mid k = 1, \ldots, N_Y\}$ denote the set of style words, where $N_Y = 4$ in our experiments, including *girl*, *boy*, *romantic* and *modern*. Let $\boldsymbol{F} = \{f_i \mid i = 1, \ldots, N_F\}$ denote the set of all the furniture categories and $\boldsymbol{S} = \{s_j \mid j = 1, \ldots, N_S\}$ denote the set of all the small object categories. N_F and N_S are 21 and 42 in our experiments, respectively. Besides the style label, each image is also annotated with colors for all the furniture categories present in the image and the small object categories it contains.

4.2 Enhancement feature encoding

Unlike [33], our approach does not differentiate living rooms and bedrooms in the dataset. We categorize the enhancement information of furniture colors and occurrence of small objects by clustering. Specifically, for a furniture category f_i, all the colors from it across the whole dataset are clustered into N_C color clusters, denoted by set $\boldsymbol{C}^i$, where $|\boldsymbol{C}^i| = N_C$. $N_C = 15$ in our experiments. For a small object category s_j, all the images that contain it form one cluster and the remaining images form the other cluster. Similarly, the two clusters are denoted by $\boldsymbol{O}^j$ where $|\boldsymbol{O}^j| = 2$. Moreover, we use C^i_k and O^j_k to indicate the k^{th} cluster for f_i and s_j, respectively.

Each cluster of either a furniture category or a small object category can be seen as a categorical variable and is represented by a one-hot encoded vector. For clarity, we use corresponding lowercase letters to denote the vectors. Specifically, we use $\boldsymbol{c}^i \in \{0,1\}^{N_C}$ to denote a color cluster for a furniture category f_i, where only one element of $\boldsymbol{c}^i$, say c^i_k is set to one to denote the k^{th} cluster in $\boldsymbol{C}^i$, i.e., C^i_k, and all the other elements are set to zero. Similarly, we use $\boldsymbol{o}^j \in \{0,1\}^2$ to denote an occurrence indicator cluster for a small object category s_j, where $o^j_1 = 1$ and $o^j_2 = 0$ when the small object category is present, and $o^j_1 = 0$ and $o^j_2 = 1$ otherwise. Each $\boldsymbol{c}^i$ and $\boldsymbol{o}^j$ is a one-hot vector. Concatenating all these vectors $\boldsymbol{c}^i$ and $\boldsymbol{o}^j$ together gives the enhancement feature vector $\boldsymbol{x} \in \{0,1\}^{N_E}$, which is of the multi-one-hot form. $N_E = N_F \cdot N_C + 2N_S$ is the length of $\boldsymbol{x}$. The components of an enhancement feature vector are illustrated in Fig. 2, assuming f_i is *chair* and s_j is *mirror*.

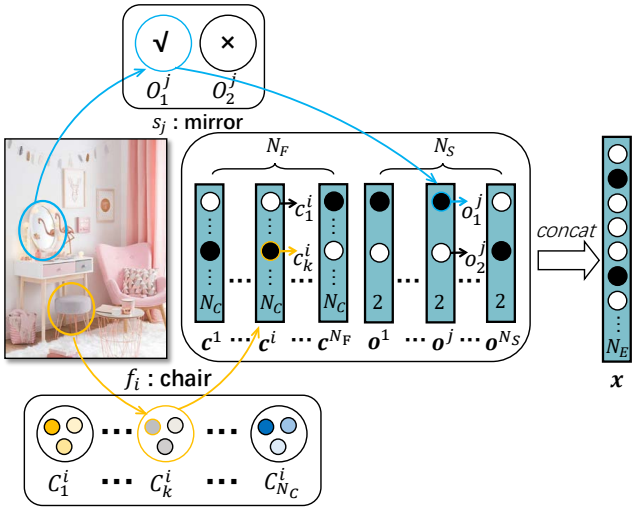


Fig. 2 Illustration of our enhancement feature encoding: one-hot vectors from N_F furniture categories and N_S small object categories are concatenated to be an enhancement feature vector $\boldsymbol{x}$ of length $N_F \cdot N_C + 2N_S$.

Style word condition. Each image in the dataset is labeled with only one style word. Therefore, during the training, we use a one-hot vector $\boldsymbol{w}$ to denote a style word. Specifically, for a style word $y_i \in \boldsymbol{Y}$, w_i is set to one while all the other elements are set to zero.

4.3 Data pre-processing

Training data augmentation. The number of images in the original dataset is not adequate for training our model. To avoid overfitting, we synthesize 2000 enhancement feature vectors for each style word $y \in \boldsymbol{Y}$ using 80% of the image dataset. The remaining data is left for testing. The synthesized dataset is denoted by **Tr-Syn**. Specifically, we first learn the probability distribution for each cluster of furniture categories conditioned on style y, $p(C^i_k | f_i, y)$, by enumerating the furniture objects of furniture category f_i belonging to color cluster C^i_k from images of style y in the original dataset. Similarly, the probability distribution for the clusters of

each small object category is also learned and denoted by $p(O_k^j|s_j, y)$. Then, in the synthesis process, given a style word y, for a furniture category f_i, a cluster C_p^i is drawn from the learned distribution $p(C_k^i|f_i, y)$ and the corresponding one-hot vector is generated by specifying $c_p^i = 1$ in $\boldsymbol{c}^i$. For a small object category s_j, O_q^j is drawn from the learned distribution $p(O_k^j|s_j, y)$ and $\boldsymbol{o}^j$ is generated in a similar way. Lastly, an enhancement feature vector given y can be obtained by first sampling for all the furniture and small object categories, and then concatenating their corresponding one-hot vectors.

Feature completion. Some images in the dataset from [33] do not contain all the furniture categories, so the color information for the absent categories is missing. However, the enhancement feature vector $\boldsymbol{x}$ is composed of one-hot vectors from the whole set of furniture categories. Thus, for a missing furniture category in an image, say f_v, we randomly sample a cluster from $\boldsymbol{C}^v$ and obtain its one-hot vector $\boldsymbol{c}^v$. The rationale behind this operation is that, when the furniture instance of a category is absent in an image, we can safely assume that the color of the missing furniture has an equal probability to be in any one of the color clusters. It guarantees that the enhancement feature vector extracted from each image is always in the multi-one-hot form regardless of how many furniture categories are present in it, which increases the robustness of our training process. Small object categories are free from this problem because, for each image, we can always assign one of the two occurrence clusters, i.e., 'presence' or 'absence', to each small object category.

4.4 Scene enhancement generation

To enhance a scene, the color cluster for each furniture category and the presence of each small object category in the scene are decoded from an enhancement feature vector $\boldsymbol{x}$. The decoding process can be seen as the reverse process of feature encoding in Fig. 2. Specifically, $\boldsymbol{x}$ is split into N_F one-hot vectors denoting color clusters for furniture categories and N_S one-hot vectors denoting the presence of small object categories.

Each furniture object of furniture category f_i in the input scene is colored with a randomly selected color from the color cluster indicated by the one-hot vector of f_i. For a small object category s_j whose cluster O_1^j (presence) is indicated by the one-hot vector of s_j, a 3D object from s_j will be selected and arranged into the input scene. We use the method in [33] to color furniture objects and arrange small objects in a 3D scene, which takes less than 1 minute. The generation of enhancement vectors from our trained generator is instant.

5 Stylistic scene enhancement GAN

As our goal is to generate scene enhancements according to one or multiple style words, we formulate our model based on a conditional GAN. Furthermore, to avoid the mode collapse issues in the traditional GANs and to improve the diversity of the generated enhancements, we specifically employ Wasserstein GAN [2] in a *conditional* sense. To deal with the sparsity and the one-hot-like features in our problem, we introduce the Gumbel-Softmax module to the generator network.

The network architecture and the training loss are detailed in Secs. 5.1 and 5.2. The training parameters and schemes are described in Sec. 5.3. The generation of a valid enhancement feature vector with the trained generator is introduced in Sec. 5.4.

5.1 Network architecture

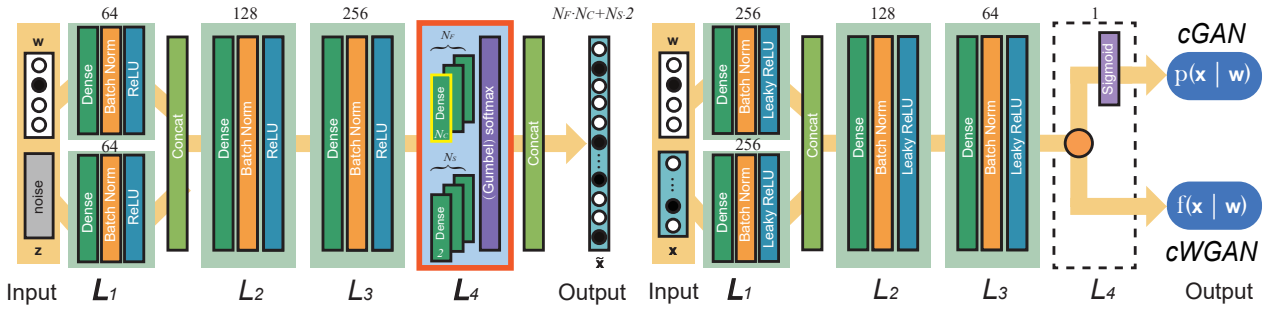
Fig. 3 demonstrates the architecture of the generator and discriminator networks used in this paper.

5.1.1 Generator

As shown in Fig. 3(a), the generator ($\mathcal{G}$) takes a noise vector $\boldsymbol{z} \sim \mathbb{P}_{\boldsymbol{z}}$, and a style vector $\boldsymbol{w} \sim \mathbb{P}_{\boldsymbol{w}}$ as the condition for the input, and outputs an enhancement feature vector $\tilde{\boldsymbol{x}}$. The two inputs are first mapped to two parallel fully connected layers of 64 units, i.e., L_{1A} and L_{1B}, collectively denoted by $\boldsymbol{L}_1$. Afterwards, the two layers in $\boldsymbol{L}_1$ are concatenated into one layer, which is mapped in sequence through two fully connected layers L_2 and L_3, with 128 and 256 units, respectively. Lastly, L_3 is mapped in parallel to N_F layers with size N_C and N_S layers with size 2 separately with full connection. We denote these layers by a layer set $\boldsymbol{L}_4$. Batch normalization and ReLU activation function are applied in sequence for all the layers except for the layers in $\boldsymbol{L}_4$. The layers in $\boldsymbol{L}_4$ are first activated by a Gumbel-Softmax [16] function and then concatenated to be the output enhancement feature, i.e., $\tilde{\boldsymbol{x}}$.

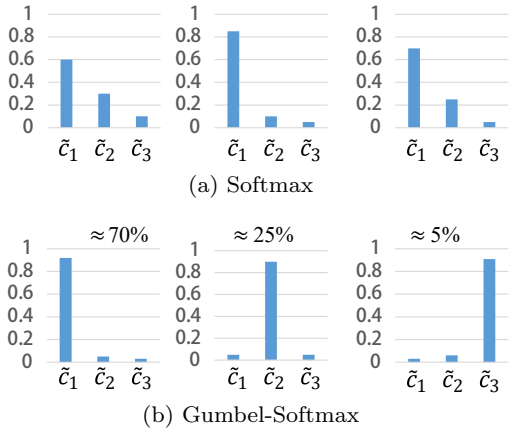
A real enhancement feature $\boldsymbol{x}$ is composed of many one-hot vectors, as detailed in Sec. 4.2. Thus, the goal of the generator is to generate a vector $\tilde{\boldsymbol{x}}$ of a similar form to $\boldsymbol{x}$. There are a few options to achieve this by activating the layers in $\boldsymbol{L}_4$, as listed below. For simplicity, we illustrate each option with a specific layer $\boldsymbol{l} \in \boldsymbol{L}_4$, which has N_C units and is highlighted by the yellow box in $\boldsymbol{L}_4$ of Fig 3(a). We denote the activation output of $\boldsymbol{l}$ by $\tilde{\boldsymbol{c}}$, which should ideally be one-hot.

Softmax. The most straightforward way is to apply the Softmax activation for $\boldsymbol{l}$. It converts $\boldsymbol{l}$ to a probability vector $\boldsymbol{\pi}$, where $\sum_i \boldsymbol{\pi}_i = 1$. $\boldsymbol{\pi}$ is a probability



(a) Generator Network (b) Discriminator Network

Fig. 3 The architecture of our generator (a) and discriminator (b). A Gumbel-Softmax module is introduced in the generator, as highlighted by the red box in (a).



(a) Softmax

(b) Gumbel-Softmax

Fig. 4 Illustration for different activation functions. Assume that the expected value for $\tilde{\boldsymbol{c}}$, $\mathbb{E}(\tilde{\boldsymbol{c}}) = [0.7, 0.25, 0.05]$. In (a), the vectors obtained by Softmax are far from one-hot. In (b), the vectors obtained by Gumbel-Softmax ($\tau = 0.1$) are not exactly but are very close to one-hot.

distribution over the color clusters of the corresponding furniture category. Fig. 4(a) shows three samples when using $\boldsymbol{\pi}$ as $\tilde{\boldsymbol{c}}$ directly where $N_C = 3$. Since these vectors $\tilde{\boldsymbol{c}}$ are very different from one-hot vectors, when concatenating them together to form $\tilde{\boldsymbol{x}}$, it will be quite easy for the discriminator to distinguish $\tilde{\boldsymbol{x}}$ from the real enhancement feature vector $\boldsymbol{x}$. As a result, the discriminator stops improving at an early stage, which further leads to inadequate training for the generator since its loss is defined by the output of the discriminator. To enforce the one-hot form, we can further apply an argmax operation after the Softmax activation. However, this operation is non-differentiable thus will block the back-propagation for training. Therefore, neither option can be used directly for activating $\boldsymbol{l}$.

Gumbel-Softmax. Unlike the above options, Gumbel-Softmax [16] activation retains the differentiability while having the ability to generate reasonable one-hot-like vectors. Specifically, we draw N_C standard Gumbel random variables in sequence and denote the set by $\{g_k \mid k = 1, \ldots, N_C\}$ where $g_k = -\log(-\log(u_k))$ with u_k drawn from $Uniform(0, 1)$ [14]. Then, $\tilde{\boldsymbol{c}}$ can be obtained by specifying each element in it as follows:

$$\tilde{\boldsymbol{c}}_k = \frac{\exp(\log(\boldsymbol{\pi}_k + g_k)/\tau)}{\sum_{i=1}^{N_C} \exp(\log(\boldsymbol{\pi}_i + g_i)/\tau)} \quad k = 1, \ldots, N_C. \tag{1}$$

$\tau \in (0, \infty)$ is a temperature hyperparameter to control how likely the generated vector is one-hot. As τ becomes larger, both $\tilde{\boldsymbol{c}}$ and the expectation of $\tilde{\boldsymbol{c}}$, $\mathbb{E}(\tilde{\boldsymbol{c}})$, get closer to a uniform vector. As τ approaches 0, $\tilde{\boldsymbol{c}}$ becomes a one-hot vector and $\mathbb{E}(\tilde{\boldsymbol{c}})$ becomes closer to $\boldsymbol{\pi}$. For example, in Fig. 4(b), when $\mathbb{E}(\tilde{\boldsymbol{c}}) = [0.7, 0.25, 0.05]$ and $\tau = 0.1$, the proportions of the three types of $\tilde{\boldsymbol{c}}$ are roughly 70%, 25% and 5%, respectively. Each type has a max value close to 1 at different dimensions. Although these vectors are fuzzy and are still not strictly one-hot, they are reasonably close to the one-hot form so can better 'fool' the discriminator than the Softmax output.

5.1.2 Discriminator

The discriminator has a similar architecture to the generator. The difference is that there is no Gumbel-Softmax module and the last layer has only one unit. As shown in Fig. 3(b), it takes a style vector condition $\boldsymbol{w}$ and an enhancement feature vector $\boldsymbol{x}$ as input, and outputs a value indicating the conformance of $\boldsymbol{x}$ to the style(s) defined by $\boldsymbol{w}$. Batch normalization and a Leaky ReLU activation function with a leak rate of 0.2 are applied for all the layers except for the last layer. For the conditional Wasserstein GAN (cWGAN) structure, the last layer is taken directly as the output, $f(\boldsymbol{x} \mid \boldsymbol{w})$. For comparison, we also construct baseline methods based on the original conditional GAN (cGAN) structure by additionally applying the *sigmoid* activation function for the last layer and obtain $p(\boldsymbol{x} \mid \boldsymbol{w})$. Both output values can be interpreted as how real the input $\boldsymbol{x}$ is conditioned on $\boldsymbol{w}$ with the exception that the former is a

probability while the latter is the Earth-Mover (EM) distance [2] between the real and 'fake' distributions.

5.2 Loss function

We reformulate the critic loss from Wasserstein GAN [2] under a style vector condition $\boldsymbol{w}$ with a gradient penalty cost introduced in [13]. The objective loss function is defined as follows:

$$\begin{aligned} L \ = & \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_{\boldsymbol{z}},\boldsymbol{w}\sim\mathbb{P}_{\boldsymbol{w}}}[\mathcal{D}(\mathcal{G}(\boldsymbol{z} \mid \boldsymbol{w}) \mid \boldsymbol{w})] - \\ & \mathbb{E}_{(\boldsymbol{x},\boldsymbol{w})\sim\mathbb{P}_{\text{data}}}[\mathcal{D}(\boldsymbol{x} \mid \boldsymbol{w})] + \\ & \lambda\mathbb{E}_{\hat{\boldsymbol{x}}\sim\mathbb{P}_{\hat{\boldsymbol{x}}}}[(||\nabla_{\hat{\boldsymbol{x}}}\mathcal{D}(\hat{\boldsymbol{x}} \mid \boldsymbol{w})||_2 - 1)^2] \end{aligned} \tag{2}$$

The first two terms account for the critic loss, which evaluates how realistic the scene enhancement features are and how well they are consistent with the style defined by the style vector $\boldsymbol{w}$. $\mathbb{P}_{\boldsymbol{z}}$ and $\mathbb{P}_{\boldsymbol{w}}$ are the prior distributions for the input noise $\boldsymbol{z}$ and the style vector $\boldsymbol{w}$, respectively. We use the uniform distribution for both distributions. The last term accounts for the gradient penalty loss. $\mathbb{P}_{\hat{\boldsymbol{x}}}$ is the implicit interpolated distribution defined by uniformly sampling along straight lines between pairs of points from the real data distribution $\mathbb{P}_{\text{data}}$, and the 'fake' data distribution $\mathbb{P}_g$ assumed by the generator $\mathcal{G}$. Specifically, $\hat{\boldsymbol{x}} = t\boldsymbol{x} + (1-t)\tilde{\boldsymbol{x}}$ with t uniformly sampled between 0 and 1, where $\tilde{\boldsymbol{x}} \sim \mathbb{P}_g$ is a scene enhancement feature generated by the generator, i.e., $\tilde{\boldsymbol{x}} = \mathcal{G}(\boldsymbol{z} \mid \boldsymbol{w})$. λ is the weight of gradient penalty and is set to 0.1 in our experiments.

For comparison, we use the following objective function for training the baseline methods using cGAN:

$$\begin{aligned} \min_G \max_D \ & \mathbb{E}_{(\boldsymbol{x},\boldsymbol{w})\sim\mathbb{P}_{\text{data}}}[\log \mathcal{D}(\boldsymbol{x} \mid \boldsymbol{w})] + \\ & \mathbb{E}_{\tilde{\boldsymbol{x}}\sim\mathbb{P}_{\tilde{\boldsymbol{x}}},\boldsymbol{w}\sim\mathbb{P}_{\boldsymbol{w}}}[\log(1 - \mathcal{D}(\tilde{\boldsymbol{x}} \mid \boldsymbol{w}))] \end{aligned} \tag{3}$$

Although we use $\mathcal{D}(\cdot)$ in both Eq. 2 for cWGAN structure and Eq. 3 for cGAN structure, it returns $f(\boldsymbol{x} \mid \boldsymbol{w})$ for the former and returns $p(\boldsymbol{x} \mid \boldsymbol{w})$ for the latter, as illustrated in Fig. 3(b).

5.3 Training

We implement our SSE-GAN in PyTorch 0.4.0 and train it on an NVIDIA Titan 1080Ti GPU. Both generator ($\mathcal{G}$) and discriminator ($\mathcal{D}$) are trained using Adam optimizer [17] with the minibatch size of 64 for 500 epochs. The learning rate of 0.0002 is used for both networks. It takes 4 hours to train SSE-GAN under this setting. During the training process, since each image is attached with only one style word, we use a one-hot style vector as the condition for both $\mathcal{G}$ and $\mathcal{D}$.

5.4 Enhancement feature generation

As our model has learned the distribution of scene enhancements conditioned on all the styles, both single or mixed stylistic enhancements can be generated with enhancement feature vectors $\tilde{\boldsymbol{x}}$ sampled from the trained generator. However, the obtained $\tilde{\boldsymbol{x}}$ is not yet strictly in multi-one-hot form. Thus, a further argmax operation is applied to $\tilde{\boldsymbol{x}}$ in a segment-wise manner regarding the definition of the enhancement feature vector in Sec. 4.2. Afterwards, the converted feature vector is decoded and used to enhance the input scene as described in Sec. 4.4.

6 Experimental results

We first evaluate the effectiveness of SSE-GAN by some evaluation metrics in Sec. 6.1. Then, we show a few scene enhancement applications enabled by SSE-GAN in Sec. 6.2. Lastly, we compare our work with an existing enhancement generation work [33] in Sec. 6.3.

6.1 Network structures

We first revisit a few sample-based evaluation metrics, and then use them for evaluating different models.

6.1.1 Evaluation metrics

We use three metrics introduced in [32], including 1NN accuracy [19], Wasserstein distance and Maximum Mean Discrepancy (MMD) [3]. To capture the dependencies between different categories, we include another metric from [4], which we refer to as the Leave-one-out category prediction accuracy distance (LOO-CPAD). For completeness, we revisit the four metrics as below.

We denote the real distribution and the generated distribution of enhancement feature vectors by $\mathbb{P}_{\text{data}}$ and $\mathbb{P}_g$, respectively. During the training, in each epoch, we generate m feature vectors using the current generator with a style vector $\boldsymbol{w}$ and m different noise vectors as input. Then, we sample another m feature vectors from the test data of the same style word. The two sets are denoted by $\boldsymbol{G} \sim \mathbb{P}_g$ and $\boldsymbol{R} \sim \mathbb{P}_{\text{data}}$, respectively. At last, the following metrics are measured for each style word separately, and the average metrics across all words are reported as the evaluation scores.

1-NN accuracy. We train a 1-nearest neighbor classifier with a set composed of $\boldsymbol{R}$ and $\boldsymbol{G}$, where $\boldsymbol{R}$ is labeled as true and $\boldsymbol{G}$ is labeled as false. Ideally, a perfect generator should generate samples that are hard to

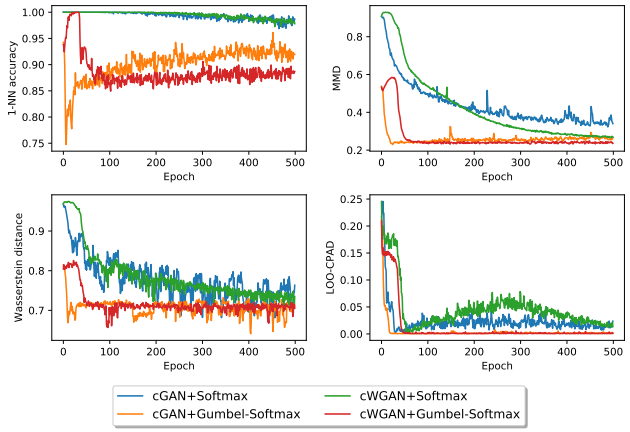


Fig. 5 Comparison of different models. 'cWGAN+Gumbel-Softmax' is used for our proposed SSE-GAN.

be differentiated from the real ones. Thus, the 1-NN accuracy of classifying all the samples in $\boldsymbol{R}$ and $\boldsymbol{G}$ should ideally be close to 50%.

The kernel MMD. It measures the dissimilarity of two distributions for a kernel function $k(\cdot,\cdot)$, which is approximated by the expectation over the distances of samples from $\boldsymbol{R}$ and $\boldsymbol{G}$:

$$\text{MMD} = \left(\mathbb{E}_{\substack{\boldsymbol{x},\boldsymbol{x}'\in\boldsymbol{R}, \\ \tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}'\in\boldsymbol{G}}} \left[k(\boldsymbol{x},\boldsymbol{x}') - 2k(\boldsymbol{x},\tilde{\boldsymbol{x}}) + k(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{x}}') \right] \right)^{\frac{1}{2}}$$

We use ℓ_1 distance for the kernel function in this paper.

Wasserstein distance. Since $\mathbb{P}_{\text{data}}$ and $\mathbb{P}_g$ are discrete distributions, their Wasserstein distance can be obtained by solving an optimal transport problem, which is referred to as Earth Mover's Distance (EMD).

Leave-one-out category prediction accuracy distance (LOO-CPAD). Each enhancement feature is composed of multiple one-hot vectors, with each representing the cluster for a furniture category or a small object category. We train a multi-class classifier for each category to predict its cluster. The dimensions from the rest categories are used as the feature. Since there are in total $N_F + N_S$ categories, an accuracy vector $\boldsymbol{a} \in \mathbb{R}^{N_F+N_S}$ can be obtained. We train two sets of classifiers separately on the training set **Tr-Syn** and $\boldsymbol{G}$, respectively, and obtain two accuracy vectors on test data, $\boldsymbol{a}_{\text{train}}$ and $\boldsymbol{a}_{\text{G}}$. The mean squared error of $\boldsymbol{a}_{\text{train}}$ and $\boldsymbol{a}_{\text{G}}$ is reported as the LOO-CPAD metric.

Except for the 1-NN accuracy (desired to be 50%), a lower score for the other three metrics indicates that the two distributions are more similar.

6.1.2 Comparison of network structures

Fig. 5 compares the four metrics during training among different combinations of cGAN or cWGAN and the activation function of Softmax or Gumbel-Softmax. As observed in Fig. 5, applying Gumbel-Softmax on either cGAN or cWGAN achieves better metrics (lower values) than applying the original Softmax. This is because Softmax produces probability vectors that are very different from the one-hot form which can be easily detected by the discriminator. With Gumbel-Softmax applied (red and orange lines), compared to cGAN, cWGAN is better on 1-NN accuracy and slightly better on MMD. For Wasserstein distance, although the two models are similar in values, the value for cWGAN is more stable than cGAN. Therefore, we use cWGAN with Gumbel-Softmax as the structure for our proposed SSE-GAN.

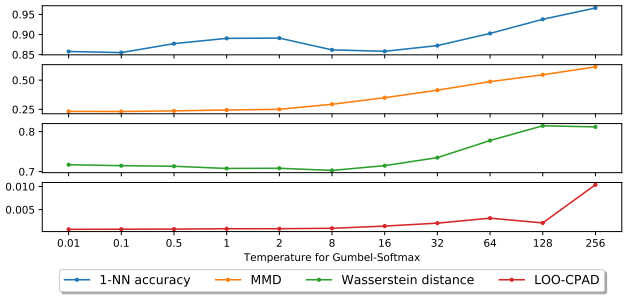


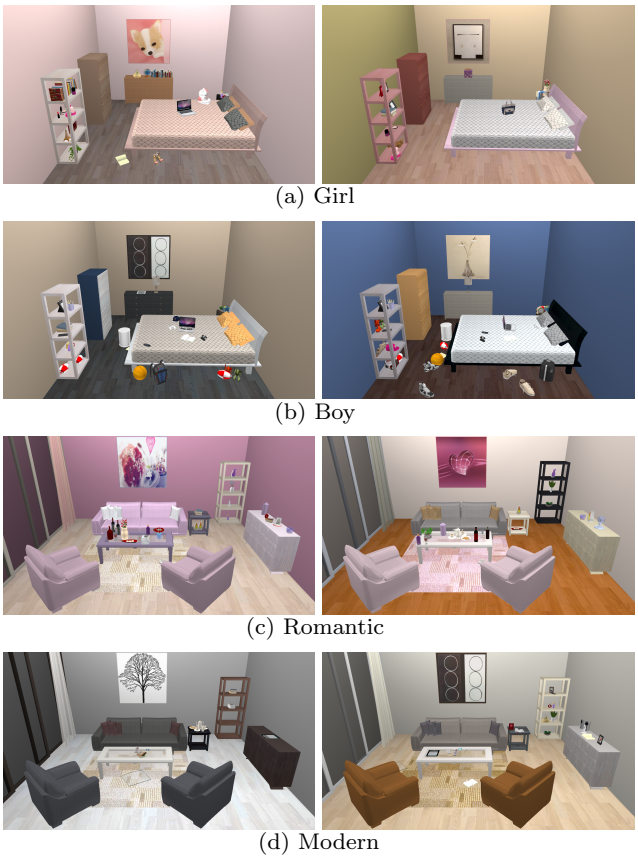
Fig. 6 Impact of the temperature parameter for Gumbel-Softmax. Temperature of 0.1 is used in this paper.

Temperature for Gumbel-Softmax. The hyperparameter of temperature τ for Gumbel-Softmax controls how likely the output vector is one-hot. We plot the metrics for SSE-GAN trained with different values of temperature. As shown in Fig. 6, our model is not very sensitive to the choice of τ as long as it is not too large. In our experiments, τ is set to 0.1.

6.2 Scene enhancement results

With the trained generator $\mathcal{G}$, various enhancements can be generated by feeding $\mathcal{G}$ a style vector and different noise vectors. Although the style vector $\boldsymbol{w}$ is one-hot during training, where each dimension corresponds to *girl, boy, romantic* and *modern*, respectively, we relax it to be a vector in $\mathbb{R}^4$ in the evaluation for mixed styles.

Single stylistic enhancements. When $\boldsymbol{w}$ is one-hot, it represents a single style. Using one-hot $\boldsymbol{w}$ and different noise vectors, we can produce various scene enhancements of single styles. Fig. 7 shows the enhancement results for a bedroom enhanced by *girl* and *boy* in the first two rows, and for a living room enhanced by *romantic* and *modern* in the last two rows. The style of the enhancements in each row is successfully possessed by the relevant small object categories and furniture colors. For example, in Fig. 7(b), the small object categories *basketball* and *gamepads* are placed in the room



(a) Girl

(b) Boy

(c) Romantic

(d) Modern

Fig. 7 Enhancements of single styles. (a) and (b) are enhancements for a bedroom of style *girl* and *boy*. (c) and (d) are for a living room of style *romantic* and *modern*.

for the *boy* style. In Fig. 7(d), black and white colors are selected for coloring the wall and sofa for the *modern* style. Moreover, diverse enhancements are enabled by providing the generator with different noise vectors, as indicated by the different enhancements shown in the first and second columns in Fig. 7.

Mixed stylistic enhancements. Mixed stylistic enhancements can be generated by a 'two-hot' style vector $\boldsymbol{w}$. Fig. 8 shows the enhancement results generated by using four different style combinations as the style condition for two indoor scenes. For each scene and each style combination, two different noise vectors are used as the input for the generator. As indicated by the two-hot style vector, two styles are simultaneously possessed by each enhancement in Fig. 8. For example, in Scene A1 of '*girl+romantic*', the furniture colors tend to be purple, which is correlated to *romantic* in our dataset, and small object categories related to *girl* such as *make up* and *doll* are placed in the room.

A user study is carried out to investigate whether the style of the generated enhancements is consistent with human perception. Specifically, 25 participants were asked to choose one out of the four style combinations for each enhancement in Fig. 8. The percentage of their choice for each enhancement is reported in Fig. 9. The bars are grouped into four according to the style combination used for generating the enhancements by our method, as indicated by the label above each group in Fig. 9. Ideally, the percentages of the style combination in the label of each group should dominate over the others. The corresponding enhancements in each group are shown in each row of Fig. 8.

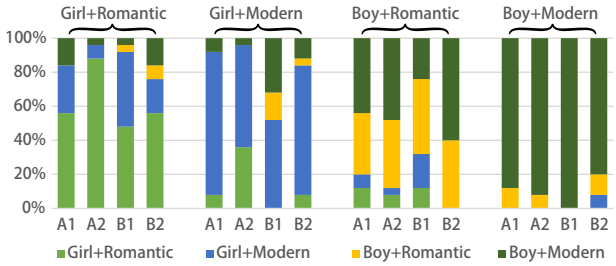


Fig. 9 Percentage of style combinations recognized by users for scene enhancements. The label above each group is the style combination used for generating the enhancements in the group.

As illustrated in Fig. 9, for the three combinations *girl+romantic*, *girl+modern* and *boy+modern*, the choice for the corresponding option consistently takes up the largest proportion among all the four options. It indicates that the enhancements generated by these style combinations are recognized as the same style combination by most participants in the user study. However, for the enhancements generated by *boy+romantic*, as shown in the third row of Fig. 8 and in the third group of Fig. 9, participants have difficulty in choosing between *boy+romantic* and *boy+modern*, which have similar proportions. We attribute this to the fact that compared to the other three combinations, *boy+romantic* is less common in our real life. Moreover, a scene possessed by *boy* style is naturally carrying some modern elements, e.g., both styles tend to have darker colors. Thus, provided a scene is recognized as *boy*, people tend to specify the style of the scene to be *boy+modern* rather than *boy+romantic* even if the scene does contain romantic elements. For example, although the enhancement for Scene B2 generated by *boy+romantic* in Fig. 8 has a purple wall, it is still recognized more as *boy+modern*, as indicated by the last bar in the third group of Fig. 9.

Interpolation for style vectors. Moreover, we investigate the ability of our trained generator to produce in-between enhancement results of two single styles by applying the linearly interpolated vectors between the two corresponding one-hot style vectors. Fig. 10(a) shows the enhancement results generated by 5 style vectors uniformly sampled along the line starting at the

Fig. 8 Mixed stylistic scene enhancements of four different style combinations for two different indoor scenes.

style vector of *girl*, $[1, 0, 0, 0]$, and ending at the style vector of *boy*, $[0, 1, 0, 0]$, with the step size of 0.25. Similarly, the in-between enhancement results for *romantic* and *modern* are shown in Fig. 10(b). As can be observed in Fig. 10, the enhancements from left to right gradually change from one style to another.

6.3 Comparison with [33]

Zhang et al. [33] proposed a framework that enhances an input scene according to a single style word. As they learned the relevance information of the enhancement factors to each word for two room types separately, their results were limited to one specific room type, i.e., either a bedroom or a living room. In our work, we do not use datasets dedicated to specific room types. Thus, our approach can generate stylistic enhancements for more complex indoor scenes regardless of the room type, as shown in Fig. 7 and Fig. 8.

Moreover, their work allows only one style word as input thus can merely generate enhancements of a single style at a time. In our work, by capturing the distribution of scene enhancements conditioned on all the styles, we are able to generate mixed stylistic enhancements, as shown in Fig. 8, as well as in-between stylistic enhancements, as shown in Fig. 10.

Lastly, even with the same input, our approach can produce more diverse results than [33]. Specifically, [33] introduced a submodular set function [24] to improve the diversity of the enhancement results. It selects a few samples from a large set of samples obtained by a Markov chain Monte Carlo sampler. When the target number of results is small, their method is still able to provide diverse results. However, when more results are demanded, their method tends to sacrifice the relevance to the input style word and choose different colors or small object categories in order to achieve the diversity. Fig. 11(a) shows the first, third and fifth enhancements selected by the submodular set function from their work using the style word *girl* for a bedroom. While the first and third enhancements (top 3) are still relevant to the word *girl*, the fifth enhancement (top 5) is less relevant as a result of improper furniture colors, e.g., the color for the wall, and the occurrence of unrelated small object categories such as *skate shoes*, as highlighted by the red boxes. In our method, diverse enhancements are generated by providing different noise vectors as the input for the trained generator. Fig. 11(b) shows three different enhancements generated by our method for the same scene as Fig. 11(a) using the same style word *girl*. They are relevant to the style word while retaining the diversity.

(a) In-between enhancement results for *girl* and *boy*.

(b) In-between enhancement results for *romantic* and *modern*.

Fig. 10 In-between enhancement results of two one-hot style vectors. (a) are the enhancement results generated by using 5 style vectors gradually changed from $[1, 0, 0, 0]$ (completely *girl*) to $[0, 1, 0, 0]$ (completely *boy*) with a step size 0.25. (b) are those generated by style vectors sampled between $[0, 0, 1, 0]$ (completely *romantic*) and $[0, 0, 0, 1]$ (completely *modern*).

(a) (b)

Fig. 11 Comparison of the diversity of our approach and [33] for the same input scene with the same style word *girl*. (a) shows the 1^{st}, 3^{rd} and the 5^{th} enhancements selected by [33]. (b) are 3 enhancements generated by our approach.

7 Discussion and Future Work

There are a few promising directions that we would like to explore in the future.

Firstly, although we take furniture colors and small object categories as the main enhancement factors while enhancing indoor scenes in this paper, our approach can be generalized to include other factors. For example, to include furniture geometry, we can first cluster 3D instances in each furniture category by their 3D descriptors such as LFD [5] or HKS [28]. Then, similar to furniture colors, furniture shapes can be represented by adopting the one-hot encoding on geometry clusters.

Secondly, our method is at the category level, so it does not support different colors for multiple 3D instances from the same furniture category. However, in real life, some styles are possessed by using certain color combinations for a specific furniture category. An attempt could be to extend our method to the instance level using the same methodology. However, it requires a sufficiently larger dataset to enable the clustering operation, which is not available at the current stage.

Thirdly, image-based feature representation can be applied to encode different aspects of enhancement styles so that convolutional neural networks can be well adopted. For example, [29] and [34] use top-view rendered images to encode the category or layout information for the furniture objects in a scene. We can extend their idea to encode information for scene styles in different channels of an image.

Lastly, the input style words are minimal due to the inadequacy of the data, which leads to very limited enhancement results. An option is to augment existing 3D indoor scene datasets [27] with stylistic annotations. Moreover, the relationship between the style words is not modeled in our approach. A possible improvement is to represent each style word with a continuous vector learned with embedding methods such as Word2Vec [22]. Therefore, users will not be limited to a few fixed words and can use other semantically similar words as input to generate stylistic scene enhancements, e.g., using *female* instead of *girl*.

8 Conclusion

In this paper, we propose the stylistic scene enhancement GAN (SSE-GAN) to generate mixed stylistic scene enhancements. To enable the learning of SSE-GAN, we introduce an effective enhancement encoding scheme, which represents different aspects of the scene enhancement by an enhancement feature vector. Moreover, to generate high fidelity scene enhancement features, we introduce a novel Gumbel-Softmax module to our generator network. The quantitative analysis

shows that our proposed SSE-GAN model is superior to the baseline models regarding four evaluation metrics. Moreover, scene enhancements of single, mixed and in-between styles are plausibly generated, which justifies that SSE-GAN has effectively modeled the distribution of scene enhancements conditioned on different styles.

Acknowledgements This work was supported by the National Natural Science Foundation of China (61373070), NSF (1813583), and Tsinghua-Kuaishou Institute of Future Media Data.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Trimble 3D warehouse (2019). URL https://3dwarehouse.sketchup.com/
2. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. arXiv preprint arXiv:1701.07875 (2017)
3. Bounliphone, W., Belilovsky, E., Blaschko, M.B., Antonoglou, I., Gretton, A.: A test of relative similarity for model selection in generative models. arXiv preprint arXiv:1511.04581 (2015)
4. Camino, R., Hammerschmidt, C., State, R.: Generating multi-categorical samples with generative adversarial networks. arXiv preprint arXiv:1807.01202 (2018)
5. Chen, D.Y., Tian, X.P., Shen, Y.T., Ouhyoung, M.: On visual similarity based 3D model retrieval. Computer Graphics Forum **22**(3), 223–232 (2003)
6. Chen, G., Li, G., Nie, Y., Xian, C., Mao, A.: Stylistic indoor colour design via bayesian network. Computers & Graphics **60**, 34–45 (2016)
7. Chen, K., Choy, C.B., Savva, M., Chang, A.X., Funkhouser, T., Savarese, S.: Text2Shape: Generating shapes from natural language by learning joint embeddings. arXiv preprint arXiv:1803.08495 (2018)
8. Chen, K., Xu, K., Yu, Y., Wang, T.Y., Hu, S.M.: Magic decorator: automatic material suggestion for indoor digital scenes. ACM Transactions on Graphics (TOG) **34**(6), 232:1–232:11 (2015)
9. Chen, X., Li, J., Li, Q., Gao, B., Zou, D., Zhao, Q.: Image2scene: transforming style of 3D room. In: Proceedings of the ACM International Conference on Multimedia, pp. 321–330 (2015)
10. Chen, Y., Lai, Y.K., Liu, Y.J.: CartoonGAN: Generative adversarial networks for photo cartoonization. In: IEEE CVPR, pp. 9465–9474 (2018)
11. Donahue, C., McAuley, J., Puckette, M.: Adversarial audio synthesis. arXiv preprint arXiv:1802.04208 (2018)
12. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS, pp. 2672–2680 (2014)
13. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. In: NIPS, pp. 5767–5777 (2017)
14. Gumbel, E.J.: Statistical theory of extreme values and some practical applications. NBS Applied Mathematics Series **33** (1954)
15. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: IEEE CVPR, pp. 5967–5976 (2017)
16. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. arXiv preprint arXiv:1611.01144 (2016)
17. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. arXiv preprint arXiv:1412.6980 (2014)
18. Liu, J., Yu, F., Funkhouser, T.: Interactive 3D modeling with a generative adversarial network. In: International Conference on 3D Vision (3DV), pp. 126–134. IEEE (2017)
19. Lopez-Paz, D., Oquab, M.: Revisiting classifier two-sample tests. arXiv preprint arXiv:1610.06545 (2016)
20. Maddison, C.J., Mnih, A., Teh, Y.W.: The concrete distribution: A continuous relaxation of discrete random variables. arXiv preprint arXiv:1611.00712 (2016)
21. Makhzani, A., Shlens, J., Jaitly, N., Goodfellow, I., Frey, B.: Adversarial autoencoders. arXiv preprint arXiv:1511.05644 (2015)
22. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)
23. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
24. Nemhauser, G.L., Wolsey, L.A., Fisher, M.L.: An analysis of approximations for maximizing submodular set functions-i. Mathematical Programming **14**(1), 265–294 (1978)
25. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
26. Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., Lee, H.: Generative adversarial text to image synthesis. arXiv preprint arXiv:1605.05396 (2016)
27. Song, S., Yu, F., Zeng, A., Chang, A.X., Savva, M., Funkhouser, T.: Semantic scene completion from a single depth image. arXiv preprint arXiv:1611.08974 (2016)
28. Sun, J., Ovsjanikov, M., Guibas, L.: A concise and provably informative multi-scale signature based on heat diffusion. Computer Graphics Forum **28**(5), 1383–1392 (2009)
29. Wang, K., Savva, M., Chang, A.X., Ritchie, D.: Deep convolutional priors for indoor scene synthesis. ACM Transactions on Graphics (TOG) **37**(4), 70:1–70:14 (2018)
30. Wu, H., Zheng, S., Zhang, J., Huang, K.: GP-GAN: Towards realistic high-resolution image blending. arXiv preprint arXiv:1703.07195 (2017)
31. Wu, J., Zhang, C., Xue, T., Freeman, B., Tenenbaum, J.: Learning a probabilistic latent space of object shapes via 3D generative-adversarial modeling. In: NIPS, pp. 82–90 (2016)
32. Xu, Q., Huang, G., Yuan, Y., Guo, C., Sun, Y., Wu, F., Weinberger, K.: An empirical study on evaluation metrics of generative adversarial networks. arXiv preprint arXiv:1806.07755 (2018)
33. Zhang, S., Han, Z., Martin, R.R., Zhang, H.: Semantic 3D indoor scene enhancement using guide words. The Visual Computer **33**(6-8), 925–935 (2017)
34. Zhang, Z., Yang, Z., Ma, C., Luo, L., Huth, A., Vouga, E., Huang, Q.: Deep generative modeling for scene synthesis via hybrid representations. arXiv preprint arXiv:1808.02084 (2018)