Scene Graph Driven Text-Prompt Generation for Image Inpainting

Tripti Shukla
Adobe Research India
trshukla@adobe.com

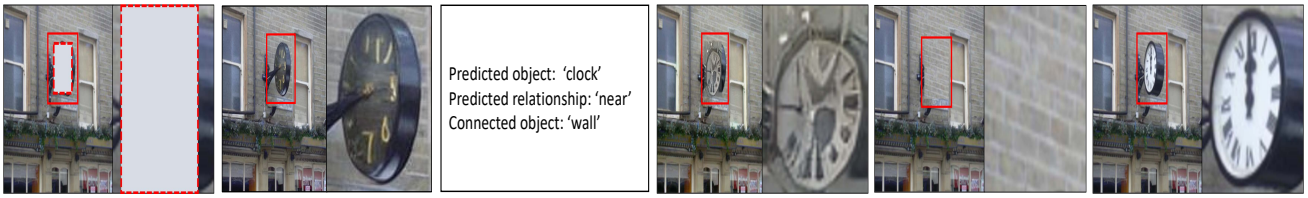
Paridhi Maheshwari
Stanford University
paridhi@stanford.edu

Rajhans Singh
Arizona State University
rsingh70@asu.edu

Ankita Shukla
Arizona State University
ashukl20@asu.edu

Kuldeep Kulkarni
Adobe Research India
kulkulka@adobe.com

Pavan Turaga
Arizona State University
pavan.turaga@asu.edu



(a) (b) (c) (d) (e) (f)

Figure 1. (a) shows the masked image as input with an object completely absent. Our approach generates visually plausible result with the new object inserted, in this case a clock, as shown in (b). It involves predicting object for the missing region ('clock') and the most probable relationship ('near') with an existing object ('wall'), shown in (c), is cast into a text-prompt into the stable diffusion model for image inpainting [23]. The same object generation by Qiu *et al.* [22] is shown in (d). State-of-the-art inpainting approach [34] fills the masked region with background texture thereby completely removing the target object, as depicted in (e). The ground truth image is shown in (f).

Abstract

Scene editing methods are undergoing a revolution, driven by text-to-image synthesis methods. Applications in media content generation have benefited from a careful set of engineered text prompts, that have been arrived at by the artists by trial and error. There is a growing need to better model prompt generation, for it to be useful for a broad range of consumer-grade applications. We propose a novel method for text prompt generation for the explicit purpose of consumer-grade image inpainting, i.e. insertion of new objects into missing regions in an image. Our approach leverages existing inter-object relationships to generate plausible textual descriptions for the missing object, that can then be used with any text-to-image generator. Given an image and a location where a new object is to be inserted, our approach first converts the given image to an intermediate scene graph. Then, we use graph convolutional networks to 'expand' the scene graph by predicting the identity and relationships of the new object to be inserted, with respect to the existing objects in the scene. The output of the expanded scene graph is cast into a textual description, which is then processed by a text-to-image generator, conditioned on the given image, to produce the final inpainted image. We conduct extensive experiments on the Visual Genome dataset, and show through qualitative and quantitative metrics that our method is superior to other methods.

1. Introduction

As AI driven content creation and content editing continues to go mainstream [15], a much needed functionality for end-users is the ability to insert new and realistic objects in a scene. Such approaches can enable new ways for content creation for augmented and virtual reality applications, as well as creating robust models for deploying field robotics [29]. The classical approach for new object insertion is to use the full knowledge of the 3D geometry of the scene, ambient lighting, shape and surface reflectance of the object, and render a new scene with this information. This is still the common workflow in current augmented reality applications. Such methods have been found limited in quality and speed due to the need to estimate all physical variables.

In the data-driven paradigm, the common approach for object insertion is broadly referred to as inpainting, which works by filling missing pixels in a region such that there is blending of the new content in the given scene context. Inpainting has a rich and long history, ranging from

methods such as diffusion-based [2, 30], model-based texture synthesis [6, 19], pyramidal/multiscale methods [7, 13], and inverse-problem frameworks [4, 18]. Data driven deep learning approaches have been used to revisit the problem, resulting in breakthroughs in visual quality using fewer assumptions on image smoothness, object geometries, or other prior knowledge. Specific methods have included convolutional methods [9, 16, 32] and GAN-based methods [20, 31]. These methods work well for filling in holes with textures, small objects, or complete the interiors of objects with visible boundaries, but lack ability to introduce new and fully detailed objects with large pixel 'footprint'.

As an alternate, newer approaches have shown how to train an end-to-end neural framework for inserting instances of specified object classes using methods such as semantic layout editing [14]. Yet, it is not clear how new objects can be 'hallucinated' under these frameworks when their classes are not explicitly specified. There is a newly emergent class of approaches, that can be referred to as 'hallucinating' approaches [12, 22], that allow one to insert 'plausible' objects, without specifying object classes manually, with high quality detail and large pixel footprint, yet not be limited by lack of 3D geometry, reflectance, and lighting estimation. These approaches use semantic information such as semantic label maps, object co-occurrence and scene-graphs, as well as image-features to learn high quality image synthesis. Qiu *et al.* [22] proposed a novel paradigm of image inpainting by expressing the image in the form of a scene graph, and casting the problem of image inpainting as one of finding, or adding, the node corresponding to the missing object. While the method is successful in adding new objects, the accuracy of the object class predicted at the missing region is far from desired. This, we believe is due to the limited scope of scene graphs that is used. Specifically, the scene graph involves only the object nodes, but the relationships between objects are completely ignored. Expanding such a graph can lead to erroneous class label or node prediction, and affects the quality of the inpainted image. In our approach, we utilize a richer representation of scene graphs defined by not only the object nodes but also the relationships between them defined by the edges of the scene graph. We cast the problem of expanding the scene graph as one of identifying the object class for the missing node, while simultaneously predicting the relationship classes between the missing node and the other existing nodes. We show that this richer representation of the scene graph allows us to predict the class for the missing object more accurately than [22], provide the object class for the missing object as well its relationship with other existing classes. While [22] uses the predicted features and the semantic label to hallucinate the object using a GAN-based architecture, we rely on the more superior stable diffusion model [23] that uses rich text prompts generated from predicted object and its relationships conditioned on the image. To this end, we obtain highly descriptive text prompts by casting the class label of the missing object, the highest probability relationship of the missing object as estimated by our approach and the corresponding existing object. Further our approach involves conditioning on these text prompts and the masked image to yield an inpainted image with a novel object inserted.

The main contribution and findings our work are summarized below.

- We propose image inpainting through object insertion as a three-stage modular process: a) scene graph expansion, b) object hallucination based on expanded scene graph, c) final image generation based on a text prompt obtained through the scene graph expansion and the masked image.

- We propose to find the object label of the missing node and its relationship with the other existing nodes through a scene graph expansion problem, that is solved using a graph convolution network. The graph expansion results in a semantic label, associated features for the plausible object and corresponding relationships of the predicted object with other objects in the scene which are used to generate text prompts.

- Our approach results in high quality object insertion as measured both by full image-level quality metrics as well as object-level quality metrics. We find through quantitative metrics like FID and IS, and human evaluation studies on the Visual Genome dataset, that our approach performs better than all baselines, including traditional image inpainting, when quality is measured tightly at the object-level, and compares favorably when quality is measured at the full image-level.

2. Related Work

Traditional inpainting methods can be categorized into diffusion-based [2, 30] and patch-based methods [1, 3, 5]. Diffusion-based methods consider the neighborhood of a missing region to synthesize new textures for the missing region, usually solved by differential equations that encode some type of smoothness in shape and texture. These methods specially fall short in incorporating high-level image semantic understanding, limiting their ability to generate appropriate object parts when large patches need to be filled in. On the other hand, patch-based methods find a matching patch in the rest of the image to fill in the missing patch. However, these methods require extensive computation to calculate the similarity between patches. Context Encoders [20] represent a breakthrough in this area, and generate images conditioned on the image-context

by embedding the image with a center patch into a low-dimensional feature space. Decoding from the feature space then generates the required patch. Post-process is then used to propagate texture information to fill in the missing patch.

Recently, VAEs and GANs have been used extensively for image inpainting. Several GAN based approaches have been developed to generate object instances. Yu *et al.* [33] develop contextual attention operation to utilize feature patches from distant areas of the image to generate pixels in the missing region. Zhang *et al.* [35] developed cascaded generators to progressively fill in the image. The method most closely related to our work is the work of Qiu *et al.* [22]. This approach trains an end-to-end network using a graph convolutional network (GCN) and GANs. The approach first transforms an image to a semantic graph that is an input to a GCN to predict the semantic information that is used by GAN to generate the missing object, even when the object is in *total absentia*. However, our method uses a richer representation of scene graphs that enables it to output both the missing object label as well as it's relationship with other objects - that is then cast into a text-prompt to stable diffusion model to generate the inpainted image. Works like Maheshwari *et al.* [17] exploit the structure in scene graphs to produce image embeddings useful for semantic image retrieval. Other related approaches include the work of Lee *et al.* [14] which uses an end-to-end trainable neural network that samples from the joint distribution of object shape and location conditioned on a semantic label map for inserting an object mask. Suin *et al.* [26] proposed a distillation based network training to improve image inpainting. Several other approaches make use of both intra-image information and learning from large datasets to improve scene authenticity and continuity.

3. Proposed Methodology

The end result we aim for is that, while filling the missing region, we want a fully fleshed out new object to be hallucinated, that also fits in the global context of the image. We are not seeking to fill in regions with similar pixels or similar patches from neighbouring regions in the image. This goal motivates our three stage approach described in detail in this section.

3.1. Scene Graph Expansion: Stages 1 and 2

Stage 1: To introduce objects that are not present in the input image, we first need an image representation that allows us to insert new objects at a semantic level. To this end, we propose to represent the given masked image as a scene graph. In [22], the scene graph is constructed as a complete graph ignoring the semantic relationship between different objects. Unlike [22], the scene-graph representation that we obtain is much richer, where in addition to the objects defined by nodes in the graph, we have semantic relationships between objects that are defined by edges between the nodes in the graph. Specifically, for the input image $\mathcal{I}$, we obtain the corresponding scene graph, G that is represented by $G = (V, E)$ - a set of vertices $V \subseteq \mathbb{V}$ and directed, labeled edges $E \subseteq \{(u, e, v) \,|\, u, v \in \mathbb{V}, u \neq v, e \in \mathbb{E}\}$ that connect pairs of objects in V. Here, $\mathbb{V}$ is the set of distinct objects found in image and $\mathbb{E}$ is the set of unique relationships. Essentially, G is made of $< subject, predicate, object >$ triples such as $< cat, on, bed >$ or $< man, driving, car >$ where objects such as 'cat', 'bed', 'man', 'car' are examples for object nodes $\in V$ and 'on', 'driving' $\in E$ are examples for relationships connecting the nodes. To obtain the scene graph as described above, we use an off-the-shelf, state-of-the-art method for scene graph generation [28]. In addition, we train an autoencoder to represent the image. The loss function for training the autoencoder is given by $L_{AE} = \frac{1}{N}\sum_{i=1}^{N} \|I_g - I_t\|_2^2$, where I_g is the ground-truth image and I_t is the output from the autoencoder.

We obtain global feature of dimension 1024 for the image by tapping into an intermediate layer of the autoencoder and then downsize it to 300 dimensions using a fully connected network. We create two additional nodes, one that encodes the image-level global feature and the other one corresponds to the missing object. Both these nodes are connected to every other node in the scene graph, G to obtain the initial expanded graph G_1. To obtain the feature representation of dimension 300 for a particular node, we simply convert the text string corresponding to the semantic label of an object using GloVe embedding [21]. Similarly, to obtain the feature representation of an edge, we obtain the GloVe embeddings for every word in the triple corresponding to the relationship defining the edge and take the average of them. Thus, every edge in the scene graph is represented by a feature dimension of 300. Compared to one-hot encoding, GloVe embeddings provide better representation because they can handle inter-class distance better, i.e., some class labels are semantically closer than others.

Stage 2: The goal of this stage is to expand the scene graph in such a way that we obtain the object feature for the missing node as well as compute the relationships of the missing node with the other existing nodes. To this end, we propose to use a graph convolutional network (GCN) [10] that takes in the initial expanded scene graph G_1 to obtain the final expanded scene graph G_e defined by features for every node and every relationship. Every layer of GCN performs message aggregation by passing information along the edges of the graph, and learns updated features for all nodes and edges. We have an object classifier head on top of the features of the missing node that outputs a probability score vector over the possible object classes and is compared with the one-hot vector for the missing object class using the cross-entropy loss. Similarly, we have a relationship classifier head on top of the features of the outgoing

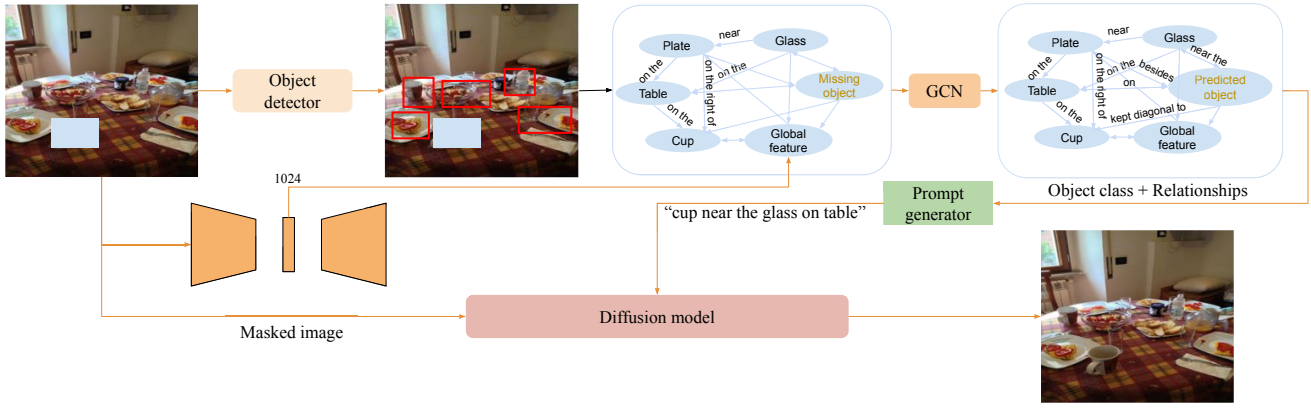


Figure 2. Overview of the proposed framework consisting of an autoencoder to derive global features, a GCN module to predict missing object and its relationships by scene graph expansion, that are cast into to a simple text prompt ('cup near the glass on table') generation that is used with a diffusion model as condition for image completion.

edges from the missing node. Each output of the relationship classifier head corresponds to one particular edge and computes a probability distribution over the possible relationship classes. To identify those edges that are not required, we simply include an additional class called 'background'. Thus, the GCN is trained with the following loss function:

$$L_{GCN} = \frac{1}{N}\sum_{i=1}^{N}\mathcal{CE}(\mathcal{H}_o(f(x_i)), o_i) + \frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{R_i}\mathcal{CE}(\mathcal{H}_r(g_j(x_i)), r_{ij}), \quad (1)$$

where x_i is the embedding of the i^{th} image in G_1, f is the series of the graph convolutional layers that operate on the graph, o_i is the one-hot representation for the ground-truth object class of the missing object, r_{ij} is the one-hot representation for the ground-truth relationship class for the j^{th} edge in the i^{th} image. $\mathcal{H}_r$ and $\mathcal{H}_o$ are the multi-layer perceptrons defining the relationship and the object head respectively, while $\mathcal{CE}$ is the cross-entropy loss function.

3.2. Text Prompt and Image Generator: Stage 3

The stage 2 provides us the object class of the predicted object, O_p along with the probability scores over all possible relationships with all existing objects in the image. We choose the relationship, R_{pe} that has the highest probability score among all the relationships predicted by $\mathcal{H}$ and the corresponding existing object, O_e. Further, we construct a text prompt given by $\mathcal{T} = \{O_p, R_{pe}, O_e\}$. The masked input image and the text prompt, $\mathcal{T}$ are fed into a stable diffusion model [23] for image inpainting that is conditioned on the image and the text prompt to produce an inpainted image as desired.

4. Experiment Setup and Results

4.1. Dataset Preparation

We use the Visual Genome [11] dataset, that has scene graph annotations for every image, and preprocess the dataset in two steps in order to make the scene-graph expansion task feasible. In the first step, in every scene graph, we only retain the nodes corresponding to object classes that occur at least 2000 times across the entire dataset. If a node is removed, we remove all the edges connected to that particular node. Next, we only retain the edges that correspond to relationship types that occur at least 500 times across the entire dataset. This helps us trim down the vocabulary and remove infrequently occurring object classes and relationship types. In the second step, we only retain the images that have at least 3 object instances, and at least one relationship. From the remaining images and their associated scene graphs (after trimming the vocabulary), we create two datasets, VG-9 and VG-20 and conduct our experiments on them. We look at the most occurring 9 object classes (out of all possible *thing* classes, i.e., excluding *stuff* classes like 'sky'), and constrain the inpainting task or scene graph expansion task of assigning the missing object to one of these 9 classes. This leads to a dataset that we call VG-9, of around 14k images where the missing object is of one of these classes. Similarly, we create another dataset for the 20 most frequent object classes, of around 86k images. In all experiments, we use 70% of the images for training and validation, and the remaining 30% for testing.

VG-9: VG-9 dataset consists of $14.36K$ images for training and $3.73K$ images for validation. Each image has one object missing corresponding to one of 9 classes. Following are the 9 classes: *bottle, clock, bowl, kite, cup, apple, backpack, doughnut, keyboard.*

VG-20: VG-20 dataset consists of $86.32K$ images for train-

ing and $22.99K$ images for validation. Each image has one object missing corresponding to one of 20 classes. Following are the 20 classes: *man, sign, car, table, chair, boat, bottle, umbrella, bag, train, airplane, clock, kite, bird, dog, horse, bus, glass, sheep, elephant*.

4.2. Network Details and Hyperparameters

Stage-1 training details: In stage 1, we train the Mask-RCNN model on our training set that takes the masked images as input. For each detected object, we construct a 1028-dimension feature vector that encodes both the semantic and spatial information. Features in first 1024 dimensions are taken directly from Mask-RCNN to account for the semantics, while features in the last four dimension are the upper-left and lower-right coordinates of the detected bounding box that accounts for the spatial information of the object. This 1028-dimensional feature vector is further fed to the Stage 2 after projecting to 300 dimension. We train the autoencoder network with learning rate of $1e-3$ for 100 epochs with a batch size of 64. We extract the first 1024-dimension feature from the Autoencoder as the global feature of the masked image and pass to Stage 2.

Stage-2 training details: For the stage 2 module, we train the GCN with image level features obtained from stage 1 and the constituent word embeddings of all relationships (edges) and objects (nodes) in the graph as input. We train this stage for 100 epochs with a batch-size of 32 with different learning rates. We get the best results at a learning rate of $3e-4$ for VG-9 dataset and $1e-4$ for VG-20. The missing object node is initialized with a 1028-dimension vector with first 1024 dimensions as the Gaussian noise and the last four dimension as the top-left and bottom-right coordinates of the masked object. The input embedding size is 1028 and the output embeddings are of 1024 dimension.

Baseline methods: For comparison we use three baselines: HVITA [22], DeepFillv2 [33] and CR-Fill [34]. HVITA [22] model is very close to ours; it also uses graph neural network to insert the object. The other two models are deep learning-based image inpainting methods. For HVITA [22] we contacted the authors for the official implementation but were not able to obtain a favourable response. We did our own implementation in PyTorch based on the details provided in the paper. We use the same training hyper-parameters for HVITA [22] as we used in our proposed method. We use the official code and training hyper-parameters recommended by the respective authors for the traditional image inpainting baselines.

4.3. Qualitative Results

In Fig 3, we show visual results for object generation in natural images by comparing our method with three baseline methods: HVITA [22], DeepFillv2 [33] and CR-Fill [34]. Column 1 in figure shows the images with masked instances provided as input to the different methods. Columns 2 shows the ground truth images for comparison. Column 3 and 4 present the images generated by DeepFillv2 and CR-Fill respectively. It can be seen from column 3 and 4 that the generated images do not visually represent the ground truth object correctly. This can be attributed to the fact that traditional inpainting methods attempt at filling the masked region with pixels from the neighbourhood and are incapable of generating semantically rich representation of different objects in natural images. In columns 5 and 6, we show the images generated by HVITA and our method respectively. It can be seen that our method produces visually plausible objects with smoother and precise shapes as compared to the baseline methods. Similarly, in Fig 4, we show the synthesized objects in natural scenes from Visual Genome dataset for HVITA baseline and our method. Columns 1, 2 and 3 present the masked image, ground truth image and the ground truth object respectively. In columns 4 and 5, we depict the generated object in the natural image and the generated object by baseline method respectively. It can be observed from columns 4 and 5 that the generated objects belongs to different class from the ground truth objects. For example, in row 1, the ground truth object is a 'kite' while the generated object by the baseline method is a 'clock'. This shows that the GCN module of the baseline method predicted incorrect class for these objects. Similarly, columns 6 and 7 present the synthesized image and the corresponding generated object by our method respectively. As seen from the figure, the generated object by our method closely resembles the ground truth object. This simple comparison shows the potential of our method in generating visually plausible and contextually correct objects in natural images.

4.4. User Study

To assess the quality of our generation and to compare it with HVITA [22], we conduct an experiment on Amazon Mechanical Turk where human workers evaluated the visual quality of object insertion in images. Within each annotation, we send the worker a randomly selected quadruplet: original image, masked object image, generated image by our method and generated image by the baseline method. We asked the worker to choose between the two generated images the better representative of original image, and which of the two generated objects better fits the scene. The results of the user study are provided in the supplementary material.

4.5. Quantitative Results

Object Class Prediction: We compare our method to HVITA [22], across metrics such as accuracy and balanced class accuracy, to demonstrate its superior capability in predicting the correct object class. Balanced accuracy is

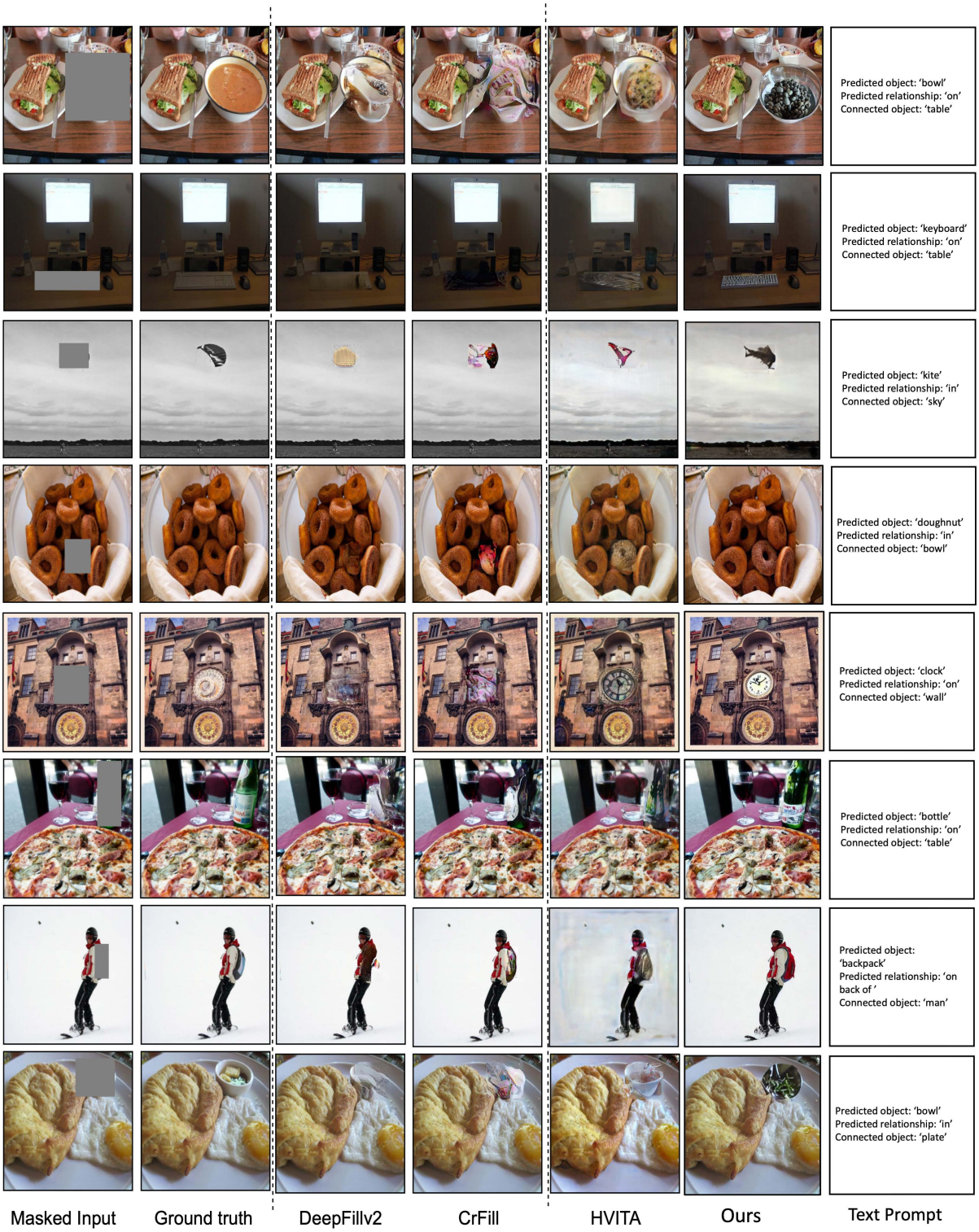


Figure 3. Visual comparison of object generation by our method with HVITA [22] and other inpainting methods. In each row, the last column represents the text-prompt that our approach gives out from stage 2 in the form of the triplet of predicted object, the maximum probability relationship and the associated existing (connected) object. This is input into the stable diffusion model for inpainting to generate much superior object insertions than all other methods.

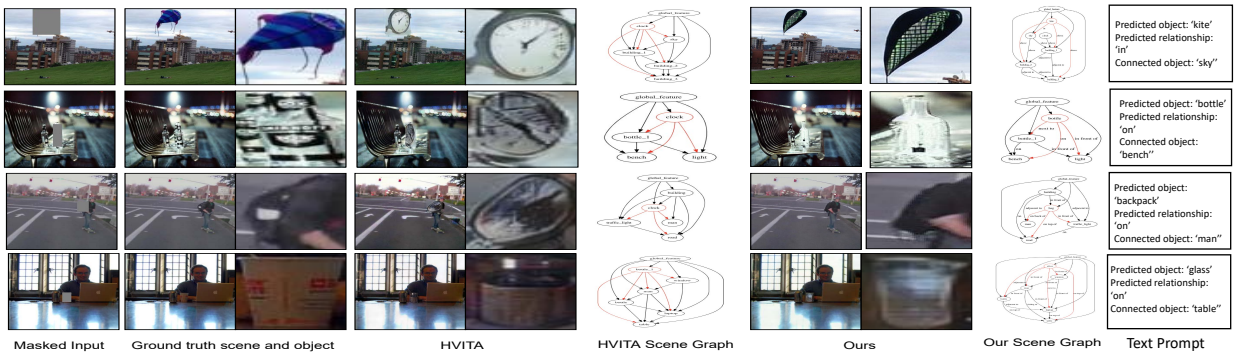


Figure 4. Visual comparison of object generation where the object predicted by our method is correct and by HVITA [22] is incorrect. This is to show the superiority of our method in object class prediction and hence also in the quality of the inpainted image. In the last column, we have provided the text prompts output by our method. Please note that while [22] leads to inaccurate inpainting, our method is able to produce highly plausible image inpaintings owing to the accurate predictions of object class and the relationships.

Method	Accuracy(%) ↑		Balanced Accu.(%) ↑	
	VG-9	VG-20	VG-9	VG-20
HVITA [22]	84.41	81.76	73.09	51.56
Ours	**84.55**	**84.47**	**73.97**	**52.95**

Table 1. The table shows the average accuracy (higher is better) and balanced accuracy (higher is better) of predicted classes by the GCN module (Stage-2) of our method and the baseline method. Scores are evaluated for VG-9 and VG-20 object classes

defined as the mean of Sensitivity (true positive rate) and Specificity (true negative rate) in a classification task. It can be seen from Table 1 that our method outperforms HVITA proposed GCN network on both metrics for VG-9 and VG-20 object classes. This is because the HVITA proposed GCN module only takes the instance and image level features as input to account for the object nodes and global semantics respectively but does not explicitly account for the relationships between the various object nodes, thus limiting the ability of the model to learn relationship importance. On the other hand, our GCN module takes into account the different relationships between the various object, thus outperforming the baseline method.

The effect of providing relationships between nodes as input to the GCN module in our method is also demonstrated by table 2 and table 3 which compares the class accuracies for a few object classes, by HVITA and the proposed method for both datasets (VG-9 and VG-20). For VG-9 dataset, out of nine object classes, our method outperforms the baseline method for six classes. Out of 20 classes, we outperform the baseline method for 13 classes. More results and visualizations are in the supplementary.

Image Inpainting Results: For this, we use the Frechét Inception Distance (FID) [8] and Inception score (IS) [25] to quantify the effectiveness of our model against the baselines. The FID and IS are the most popular metrics used in the generative model literature. The IS uses an inception-v3 model [27] pre-trained on the ImageNet dataset [24] and calculates the KL divergence between the conditional distribution (from the generated images) and the marginal distribution. The inception score is higher if the generated objects are sharp in the image. One of the drawbacks of IS is that it does not compare the distribution of the generated images with the real ones. FID solves this problem by assuming Gaussian distribution of inception features and computes the Frechét distance between the features of real images and the generated images.

In our quantitative analysis, we do not use metric like SSIM and PSNR that directly compares the generated image with the corresponding ground truth image. Natural objects belonging to the same class can occur with varying shapes and color textures. Generators trained with GAN loss are not restricted to generating the same color texture and mainly focus on generating a realistic-looking image. Hence, the proposed model generates object images with different textures and colors, unlike the ground truth object.

Fig 5 shows the FID score of our proposed model and the baselines evaluated at different patch sizes extracted from the image. In this figure, $\alpha = 0$ means that we only compare the generated object images with the ground-truth object distribution, and $\alpha = 1.0$ means the whole image is used for evaluation. In any natural scene, area occupied by any object is much smaller than the entire image. The FID score computed at $\alpha = 1.0$ gives significantly less weightage to the generated region in the image. This can be seen in Fig 5, at $\alpha = 0.0$, our model performs significantly better than the baselines. Hence, we evaluate all methods only on the generated object level.

In tables 4 and 5, we observe that our proposed model

Object	apple	backpack	bottle	bowl	clock	cup	doughnut	keyboard	kite
HVITA [22]	**64.88**	60.57	88.19	75.97	92.62	55.76	**59.13**	74.25	**95.42**
Ours	62.35	**64.94**	**90.15**	**79.48**	**93.79**	**57.22**	48.56	**74.78**	94.42

Table 2. The table shows the class accuracy percentage (higher is better) for some object categories in VG-9 dataset

Object	man	sign	car	table	chair	boat	bottle	umbrella	bag	train	airplane	clock	kite	bird	dog	horse	bus	glass	sheep	elephant
HVITA [22]	92.67	78.81	76.13	**75.24**	**54.25**	**47.95**	**49.83**	39.54	31.14	51.95	47.14	43.91	45.49	**43.22**	32.84	41.45	**44.08**	49.09	36.90	**49.44**
Ours	**93.35**	**85.43**	**80.33**	74.99	48.71	47.33	49.42	**46.82**	**36.11**	**52.06**	**51.76**	**48.97**	**45.68**	40.01	**38.78**	**44.07**	37.95	**49.16**	**38.76**	49.14

Table 3. The table shows the class accuracy percentage (higher is better) for some object categories in VG-20 dataset

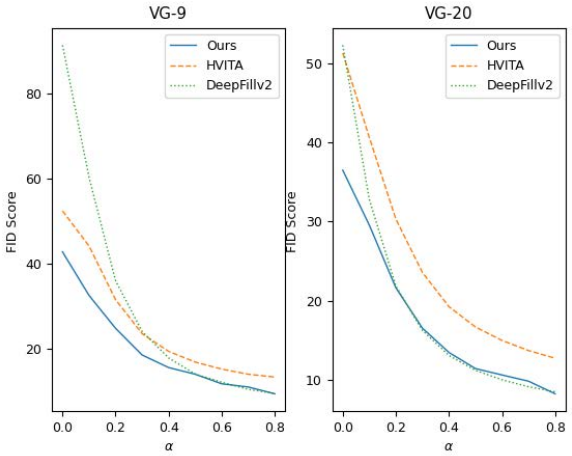


Figure 5. The figure shows FID scores (lower is better) evaluated at different patch sizes on VG-9 and VG-20 datasets. $\alpha = 0$ means that the patch tightly bounds the object of interest, and $\alpha = 1.0$ implies that the whole image is used for evaluation.

Method	FID Score ↓	Inception Score ↑
DeepFillv2 [33]	91.35	4.94
CR-Fill [34]	100.12	4.78
HVITA [22]	52.37	6.81
Ours	**42.72**	**7.56**

Table 4. This table shows the FID score (lower is better) and Inception score (higher is better) of different models on the VG-9 dataset. Scores are evaluated on the generated object patch images of the test set.

performs significantly better on both metrics (FID and IS) and both datasets (VG-9 and VG-20). We also observe that the FID and IS value for the image inpainting methods – CR-Fill [34] and DeepFillv2 [33] – are worse as compared to the proposed method and HVITA [22]. This large performance gap indicates that the inpainting methods do not generate objects, simply trying to generate realistic

Method	FID Score ↓	Inception Score ↑
DeepFillv2 [33]	52.25	5.51
CR-Fill [34]	88.63	4.92
HVITA [22]	51.26	6.08
Ours	**36.47**	**8.42**

Table 5. This table shows the FID score (lower is better) and Inception score (higher is better) of different models on the VG-20 dataset. Scores are evaluated on the generated object patch images of the test set.

background, as seen in our qualitative results. In tables 4 and 5, we also observe that our proposed method outperforms HVITA [22] by a significant margin on FID score. This indicates the effectiveness of using a richer scene graph representation as done in our method.

5. Conclusion

We proposed a scene aware object insertion method using object relationships. To this end, we proposed a three step approach – we first generate a scene graph that is expanded with label and node relationships for a plausible object. The expanded scene graph is processed by a GCN module to predict object class label and relationships, that are processed by a text-to-image generator along with the masked image to generate the final results. Thus we provide a principled way of generating a text-prompt that is composed of two objects and their associated relationship that can be exploited by and plugged into any text-to-image generator. The rich representation of scene graph enables predicting the class of the missing object and it's relationships between other objects in the image very accurately. Our approach presents a way to generate appropriate text prompts so that one can leverage the latest advances in text-to-image synthesis, in a modular way. This in turn results in the quality of generated image with inserted object to be better than existing methods both in terms of image-level as well as object-level quality metrics.

References

[1] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009.

[2] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, page 417–424, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[3] Marcelo Bertalmio, Luminita Vese, Guillermo Sapiro, and Stanley Osher. Simultaneous structure and texture image inpainting. *IEEE transactions on image processing*, 12(8):882–889, 2003.

[4] Raymond H. Chan, Junfeng Yang, and Xiaoming Yuan. Alternating direction method for image inpainting in wavelet domains. *SIAM Journal on Imaging Sciences*, 4(3):807–826, 2011.

[5] Antonio Criminisi, Patrick Pérez, and Kentaro Toyama. Region filling and object removal by exemplar-based image inpainting. *IEEE Transactions on image processing*, 13(9):1200–1212, 2004.

[6] A.A. Efros and T.K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1033–1038 vol.2, 1999.

[7] David J. Heeger and James R. Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, page 229–238, 1995.

[8] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[9] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Trans. Graph.*, 36(4), jul 2017.

[10] Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

[11] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017.

[12] Kuldeep Kulkarni, Tejas Gokhale, Rajhans Singh, Pavan K. Turaga, and Aswin C. Sankaranarayanan. Halluci-net: Scene completion by exploiting object co-occurrence relationships. In *AI for Content Creation (held in conjunction with CVPR)*, 2021.

[13] Olivier Le Meur, Mounira Ebdelli, and Christine Guillemot. Hierarchical super-resolution-based inpainting. *IEEE Transactions on Image Processing*, 22(10):3779–3790, 2013.

[14] Donghoon Lee, Sifei Liu, Jinwei Gu, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Context-aware synthesis and placement of object instances. *Advances in Neural Information Processing Systems*, 31, 2018.

[15] Huan Ling, Karsten Kreis, Daiqing Li, Seung Wook Kim, Antonio Torralba, and Sanja Fidler. Editgan: High-precision semantic image editing. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[16] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[17] Paridhi Maheshwari, Ritwick Chaudhry, and Vishwa Vinay. Scene graph embeddings using relative similarity supervision, 2021.

[18] Julien Mairal, Michael Elad, and Guillermo Sapiro. Sparse representation for color image restoration. *IEEE Transactions on Image Processing*, 17(1):53–69, 2008.

[19] R. Paget and I.D. Longstaff. Texture synthesis via a noncausal nonparametric multiscale markov random field. *IEEE Transactions on Image Processing*, 7(6):925–931, 1998.

[20] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition CVPR*, pages 2536–2544, 2016.

[21] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[22] Jiayan Qiu, Yiding Yang, Xinchao Wang, and Dacheng Tao. Hallucinating visual instances in total absentia. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 264–282. Springer, 2020.

[23] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

[25] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

[26] Maitreya Suin, Kuldeep Purohit, and AN Rajagopalan. Distillation-guided image inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2481–2490, 2021.

[27] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[28] Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. Unbiased scene graph generation from biased training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3716–3725, 2020.

[29] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[30] Andrew Witkin and Michael Kass. Reaction-diffusion textures. *SIGGRAPH Comput. Graph.*, 25(4):299–308, jul 1991.

[31] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtman, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition CVPR*, pages 4076–4084. IEEE Computer Society.

[32] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5505–5514, June 2018.

[33] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018.

[34] Yu Zeng, Zhe Lin, Huchuan Lu, and Vishal M Patel. Cr-fill: Generative image inpainting with auxiliary contextual reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14164–14173, 2021.

[35] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.