Tell, Draw, and Repeat: Generating and Modifying Images Based on Continual Linguistic Instruction

Alaaeldin El-Nouby[1,4,*] Shikhar Sharma[2] Hannes Schulz[2] Devon Hjelm[2,3,5]
Layla El Asri[2] Samira Ebrahimi Kahou[2] Yoshua Bengio[3,5,6] Graham W. Taylor[1,4,6]
[1] University of Guelph [2] Microsoft Research [3] Montreal Institute for Learning Algorithms
[4] Vector Institute for Artificial Intelligence [5] University of Montreal [6] Canadian Institute for Advanced Research

Abstract

Conditional text-to-image generation is an active area of research, with many possible applications. Existing research has primarily focused on generating a single image from available conditioning information in one step. One practical extension beyond one-step generation is a system that generates an image iteratively, conditioned on ongoing linguistic input or feedback. This is significantly more challenging than one-step generation tasks, as such a system must understand the contents of its generated images with respect to the feedback history, the current feedback, as well as the interactions among concepts present in the feedback history. In this work, we present a recurrent image generation model which takes into account both the generated output up to the current step as well as all past instructions for generation. We show that our model is able to generate the background, add new objects, and apply simple transformations to existing objects. We believe our approach is an important step toward interactive generation. Code and data is available at: `https://www.microsoft.com/en-us/research/project/generative-neural-visual-artist-geneva/`.

1. Introduction

Vision is one of the most important ways in which humans experience, interact with, understand, and learn about the world around them. Intelligent systems that can generate images and video for human users have a wide range of applications, from education and entertainment to the pursuit of creative arts. Such systems also have the potential to serve as accessibility tools for the physically impaired; many modern and creative works are now generated or edited using digital graphic design tools, and the complexity of these tools can lead to inaccessibility issues, particularly with people with insufficient technical knowledge or resources. A system that can follow speech- or text-based instructions and then perform a corresponding image editing task could improve accessibility substantially. These benefits can easily extend to other domains of image generation such as gaming, animation, creating visual teaching material, etc. In this paper, we take a step in this exciting research direction by introducing the *neural visual artist* task.

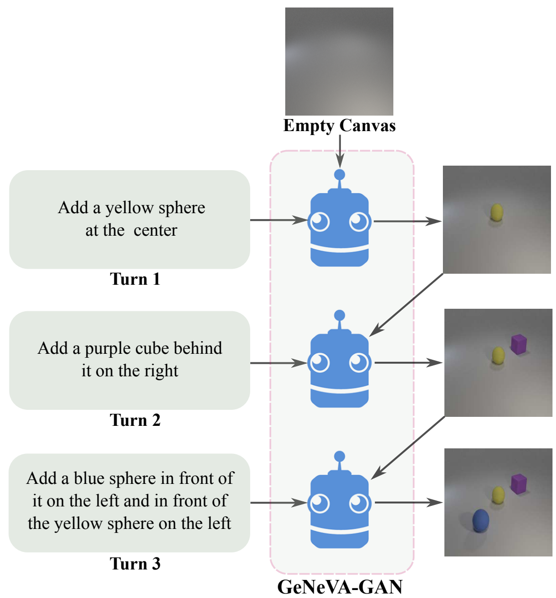


Figure 1. We present the Generative Neural Visual Artist (GeNeVA) task. Starting from an empty canvas, a *Drawer* (GeNeVA-GAN) iteratively constructs a scene based on a series of instructions and feedback from a *Teller*.

Conditional generative models allow for generation of images from other input sources, such as labels [1] and dialogue [2]. Image generation conditioned on natural language is a difficult yet attractive goal [3, 4, 5, 6]. Though these models are able to produce high quality images for simple datasets, such as birds, flowers, furniture, etc., good caption-conditioned generators of complex datasets, such as Microsoft Common Objects in Context (MS COCO) [7] are nonexistent. This lack of good generators may be due to the limited information content of captions, which are not rich enough to describe an entire image [2]. Combining ob-

*Work was performed during an internship with Microsoft Research.

ject annotations with the intermediate steps of generating bounding boxes and object masks before generating the final images can improve results [5].

Instead of constructing images given a caption, we focus on learning to *iteratively* generate images based on continual linguistic input. We call this task the Generative Neural Visual Artist (GeNeVA), inspired by the process of gradually transforming a blank canvas to a scene. Systems trained to perform this task should be able to leverage advances in text-conditioned single image generation.

1.1. GeNeVA Task and Datasets

We present an example dialogue for the GeNeVA task in Figure 1, which involves a Teller giving a sequence of linguistic instructions to a Drawer for the ultimate goal of image generation. The Teller is able to gauge progress through visual feedback of the generated image. This is a challenging task because the Drawer needs to learn how to map complex linguistic instructions to realistic objects on a canvas, maintaining not only object properties but relationships between objects (e.g., relative location). The Drawer also needs to modify the existing drawing in a manner consistent with previous images and instructions, so it needs to remember previous instructions. All of these involve understanding a complex relationship between objects in the scene and how those relationships are expressed in the image in a way that is consistent with all instructions given.

For this task, we use the synthetic Collaborative Drawing (CoDraw) dataset [8], which is composed of sequences of images along with associated dialogue of instructions and linguistic feedback (Figure 2). Also, we introduce the Iterative CLEVR (i-CLEVR) dataset (Figure 4), a modified version of the Compositional Language and Elementary Visual Reasoning (CLEVR) [9] dataset, for incremental construction of CLEVR scenes based on linguistic instructions. Offloading the difficulty of generating natural images by using two well-studied synthetic datasets allowed us to better assess progress on the GeNeVA task and improve the iterative generation process. While photo-realistic images will undoubtedly be more challenging to work with, our models are by no means restricted to synthetic image generation. We expect that insights drawn from this setting will be crucial to success in the natural image setting.

The most similar task to GeNeVA is the task proposed by the CoDraw [8] authors. They require a model to build a scene by placing the clip art images of the individual objects in their correct positions. In other words, the model predictions will be in coordinate space for their task, while for a model for the GeNeVA task they will be in pixel space. Natural images are in scope for the GeNeVA task, where Generative Adversarial Networks (GANs) are currently state-of-the-art. Non-pixel-based approaches will be limited to placing pre-segmented specific poses of objects. For such approaches, it will be extremely difficult to obtain a pre-segmented set of all possible poses of all objects e.g., under different lighting conditions. Additionally, a pixel-based model does not necessarily require object-labels so it can easily scale without such annotation.

1.2. Contributions

Our primary contributions are summarized as follows:

- We introduce the GeNeVA task and propose a novel recurrent GAN architecture that specializes in plausible modification of images in the context of an instructional history.
- We introduce the i-CLEVR dataset, a sequential version of CLEVR [9] with associated linguistic descriptions for constructing each CLEVR scene, and establish a baseline for it.
- We propose a relationship similarity metric that evaluates the model's ability to place objects in a plausible position compatible with the instructions.
- We demonstrate the importance of iterative generation for complex scenes by showing that our approach outperforms the non-iterative baseline.

Our experiments on the CoDraw and i-CLEVR datasets show that our model is capable of generating images that incrementally build upon the previously generated images and follow the provided instructions. The model is able to learn complex behaviors such as drawing new objects, moving objects around in the image, and re-sizing these objects. In addition to reporting qualitative results, we train an object localizer and measure precision, recall, F1 score, and our proposed relational similarity metric by comparing detections on ground-truth vs. generated images.

2. Related Work

GANs [10] represent a powerful family of generative models whose benefits and strengths extend to conditional image generation. Several approaches for conditioning exist, such as conditioning both the generator and discriminator on labels [1], as well as training an auxiliary classifier as part of the discriminator [11]. Closer to GeNeVA text-based conditioning, Reed et al. [3] generate images conditioned on the provided captions. Zhang et al. [4] proposed a two-stage model called StackGAN, where the first stage generated low resolution images conditioned on the caption, and the second stage generated a higher resolution image conditioned on the previous image and the caption. Hong et al. [5] proposed a three-step generation process where they use external segmentation and bounding box annotation for MS COCO to first generate bounding boxes, then a mask for the object, and then the final image. Building upon StackGAN, AttnGAN [6] introduced an attentional generator network that enabled the generator to synthesize different spatial locations in the image, conditioned on an attention mecha-

Turn 1
Teller: top left corner big sun, orange part cut. right side far right medium apple tree. i see 4 apples

Drawer: ok ready

Turn 2
Teller: left side girl big size, running, facing right. head above horizon.

Drawer: ok

Turn 3
Teller: covering the tree, on the right side of the scene is a boy, kicking, facing left. head on green part. big size, black glasses. kicking ball.

Drawer: ok

Turn 4
Teller: make tree a size bigger, move it up and left a bit. boys hand covers trunk.

Drawer: ok

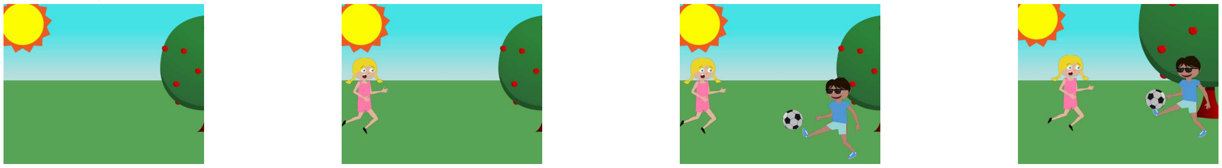


Figure 2. An example from the CoDraw [8] dataset. Each example from the dataset involves a conversation between a Teller and a Drawer. The Teller has access to a final image and has to iteratively provide text instructions and feedback to the Drawer to guide them to draw the same image. The Drawer updates the image on receiving instructions or feedback. In the original CoDraw setup, the Drawer predicted the position and attributes of objects to compose a scene. In GeNeVA, we task systems with generating the images directly in pixel space.

nism over words in the caption. It also introduced an image-text similarity module which encouraged generating images more relevant to the provided caption.

Departing from purely caption data, Sharma et al. [2] proposed a non-iterative model called ChatPainter that generates images using dialogue data. ChatPainter conditions on captions from MS COCO and a Recurrent Neural Network (RNN)-encoded dialogue relevant to the caption (obtained from the Visual Dialog (VisDial) [12] dataset) to generate images. The authors showed that the question answering-based dialogue captured richer information about the image than just the caption, which enabled ChatPainter to generate superior images compared to using captions alone. Since the VisDial dialogues were collected separately from the MS COCO dataset, there are no intermediate incremental images for each turn of the dialogue. The model, thus, only reads the entire dialogue and generates a single final image, so this setup diverges from a real-life sketch artist scenario where the artist has to keep making changes to the current sketch based on feedback.

There has also been recent work in performing recurrent image generation outside of text-to-image generation tasks. Yang et al. [13] perform unsupervised image generation in recursive steps, first generating a background, subsequently conditioning on it to generate the foreground and the mask, and finally using an affine transformation to combine the foreground and background. Lin et al. [14] tackle the image compositing task of placing a foreground object on a background image in a natural location. However, this approach is limited to fixed object templates, and instead of generating images directly, the model recursively generates parameters of transformations to continue applying to an object template until the image is close enough to natural image manifold. Their approach also does not modify existing objects in the image. Both of these approaches aim to generate a single final image without incorporating any external feedback. To the best of our knowledge, the proposed model is the first of its kind that can recursively generate and modify intermediate images based on continual text instructions such that every generated image is consistent with past instructions.

3. Methods

In this section, we describe a conditional recurrent GAN model for the GeNeVA task. An overview of the model architecture is shown in Figure 3.

3.1. Model

During an n-step interaction between a Teller and a Drawer, the Teller provides a drawing canvas x_0 and a sequence of instructions $Q = (q_1, \ldots, q_n)$. For every turn in the conversation, a conditioned generator G outputs a new image

$$\widetilde{x}_t = G(z_t, h_t, f_{G_{t-1}}), \tag{1}$$

where z_t is a noise vector sampled from a normal distribution $\mathcal{N}(0,1)$ of dimension N_z. G is conditioned on two variables, h_t and $f_{G_{t-1}}$, where h_t is a context-aware condition and $f_{G_{t-1}}$ is context-free.

The context-free condition $f_{G_{t-1}} = E_G(\widetilde{x}_{t-1})$ is an encoding of the previously generated image $\widetilde{x}_{t-1}$ using an encoder E_G, which is a shallow Convolutional Neural Network (CNN). Assuming square inputs, the encoder produces low resolution feature maps of dimensions $(K_g \times K_g \times N_g)$.

The context-aware condition h_t needs to have access to the conversation history such that it can learn a better encoding of the instruction in the context of the conversation history up to time $t-1$.

Each instruction q_t is encoded using a bi-directional Gated Recurrent Unit (GRU) on top of GloVe word embeddings [15]. This instruction encoding is denoted by d_t.

We formulate h_t as a recursive function R, which takes the instruction encoding d_t as well as the previous condition h_{t-1} as inputs. We implement R with a second GRU, which yields h_t with dimension N_c:

$$h_t = R(d_t, h_{t-1}). \tag{2}$$

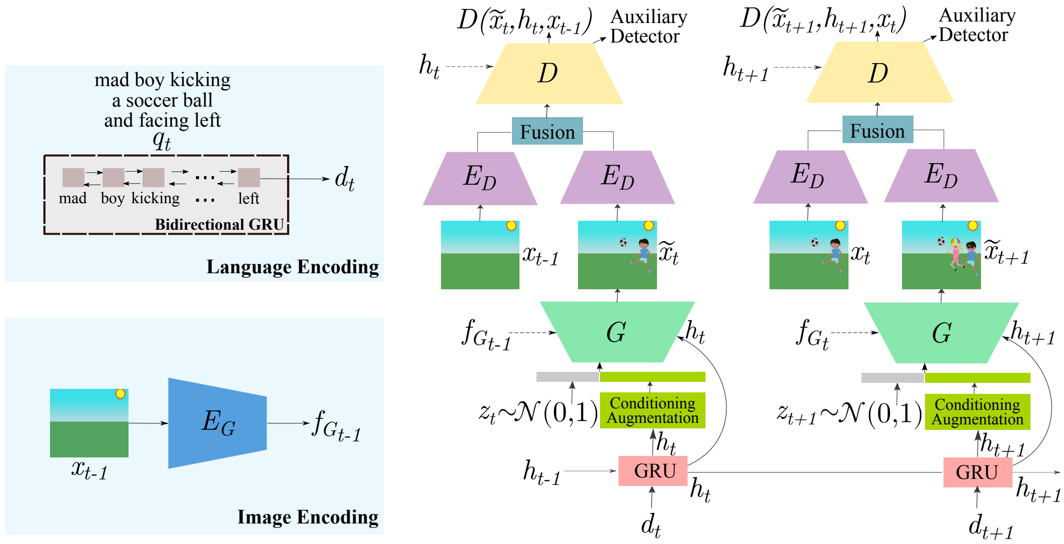


Figure 3. Overview of the GeNeVA-GAN architecture. For each time-step t, instruction q_t is encoded into d_t using a bi-directional GRU. The previous time-step generated image $\tilde{x}_{t-1}$ (teacher-forcing at training time with ground truth x_{t-1}) is encoded into $f_{G_{t-1}}$ using E_G. A GRU outputs a context-aware condition h_t as a function of d_t and the previous condition h_{t-1}. The generator G generates an image $\tilde{x}_t$ conditioned on h_t and $f_{G_{t-1}}$. $f_{G_{t-1}}$ is concatenated to feature maps from G with the same spatial dimensions while h_t is used as the input for conditional batch normalization. The image from the current time-step (ground truth x_t or generated $\tilde{x}_t$) and the previous time-step ground-truth image are encoded using E_D. The features from both images are fused and then passed as input to a discriminator D. Finally, D is conditioned using the context-aware condition h_t. An auxiliary objective of detecting all the objects in the scene is also added to D.

The context-free condition $f_{G_{t-1}}$ represents the prior given to the model by the most recently generated image (i.e. a representation of the current canvas). On the other hand, the context-aware condition h_t represents the modifications the Teller is describing in the new image. In our model, the context-aware condition is concatenated with the noise vector z_t after applying conditioning augmentation [4], as shown in Figure 3. Similar to Miyato and Koyama [16], it is also used in applying conditional batch normalization to all of the generator's convolutional layers. The context-free condition $f_{G_{t-1}}$ is concatenated with the feature maps from the generator's intermediate layer L_{f_G} which has the same spatial dimensions as $f_{G_{t-1}}$.

Since we are modeling iterative modifications of images, having a discriminator D that only distinguishes between real and generated images at each step will not be sufficient. The discriminator should also identify cases where the image is modified incorrectly with respect to the instruction or not modified at all. To enforce this, we introduce three modifications to the discriminator. First, an image encoder E_D is used to encode the current time step image (real or generated) *and the previous time-step ground-truth image* as shown in Figure 3. The output feature maps of dimensions ($K_d \times K_d \times N_d$) are passed through a fusion layer. We experiment with element-wise subtraction and concatenation of feature maps as different options for fusion. The fused features are passed through a discriminator D. Passing a fused representation of both the current and the previous images to the discriminator encourages it to focus on the quality of the modifications, not only the overall image quality. This provides a better training signal for the generator. Additionally, the context-aware condition h_{t-1} is used as a condition for D through projection similar to [16].

Second, for the discriminator loss, in addition to labelling real images as positive examples and generated images as negative examples, we add a term for the combination of [real image, wrong instruction], similar to Reed et al. [3]. Finally, we add an auxiliary objective [11] of detecting all objects in the scene at the current time step.

The generator and discriminator are trained alternately to minimize the adversarial hinge loss [17, 18, 19]. The discriminator minimizes

$$L_D = L_{D_{\text{real}}} + \frac{1}{2}(L_{D_{\text{fake}}} + L_{D_{\text{wrong}}}) + \beta L_{\text{aux}}, \tag{3}$$

where

$$\begin{aligned} L_{D_{\text{real}}} &= -\mathbb{E}_{(x_t, c_t) \sim p_{\text{data}(0:T)}}[\min(0, -1 + D(x_t, c_t))] \\ L_{D_{\text{wrong}}} &= -\mathbb{E}_{(x_t, \hat{c}_t) \sim p_{\text{data}(0:T)}}[\min(0, -1 - D(x_t, \hat{c}_t))] \\ L_{D_{\text{fake}}} &= -\mathbb{E}_{z_t \sim \mathcal{N}, c_t \sim p_{\text{data}(0:T)}}[\min(0, -1 - D(G(z_t, \tilde{c}_t), c_t))], \end{aligned}$$

with $\tilde{c}_t = \{h_t, f_{G_{t-1}}\}$ and $c_t = \{h_t, x_{t-1}\}$. Finally, $\hat{c}_t$ is the same as c_t but with a wrong instruction and T is the length of the instruction sequence Q.

The loss function for the auxiliary task is a binary cross entropy over all the N possible objects at that time step,

$$L_{\text{aux}} = \sum_{i=0}^{N} -\left(y_i \log(p_i) + (1 - y_i) \log(1 - p_i)\right),$$

where y_i is a binary label for each object indicating whether it is present in the scene at the current time step. Note that we do not index the loss with t to simplify notation. A linear layer of dimension N is added to the last discriminator layer before applying projection conditioning with h_t. A sigmoid function is applied to each of the N outputs yielding p_i, the model detection prediction for object i.

The generator loss term is

$$L_G = -\mathbb{E}_{z_t \sim p_z, c_t \sim p_{data}(0:T)} D(G(z_t, \tilde{c}_t), c_t) + \beta L_{\text{aux}} \quad (4)$$

Additionally, to help with training stability, we apply zero-centered gradient penalty regularization to the discriminator's parameters Φ on the real data alone with weighting factor γ as suggested by Mescheder et al. [20],

$$\text{GPReg}(\Phi) = \frac{\gamma}{2}\mathbb{E}_{p_{D(x)}}[\|\nabla D_\Phi(x)\|^2]. \quad (5)$$

3.2. Implementation Details

The network architecture for the generator and discriminator follows the ResBlocks architecture as used by Miyato and Koyama [16]. Following SAGAN [19], we add a self-attention layer to the intermediate layers with spatial dimensions of 16×16 for the discriminator and the generator. We use spectral normalization [18] for all layers in the discriminator.

For the training dynamics, the generator and discriminator parameters are updated every time step, while the parameters of E_G, R and the text encoder are updated every sequence. The text encoder and the network R are trained with respect to the discriminator objective only.

We add layer normalization [21] to the text encoding GRU, as well as the the GRU implementing R. We add batch normalization [22] to the output of the image encoder E_G. We found that adding these normalization methods was important for gradient flow to all modalities.

For training, we used teacher forcing by using the ground truth images x_{t-1} instead of the generated image $\widetilde{x}_{t-1}$, but we use $\widetilde{x}_{t-1}$ during test time. We use the Adam [23] optimizer for the GAN, with learning rates of 0.0004 for the discriminator and the 0.0001 for the generator, trained with an equal number of updates. We use Adam as well for the text encoder with learning rate of 0.003, and for the GRU with learning rate of $3 \cdot 10^{-4}$.

In our experiments the following hyper-parameters worked the best, $N_z = 100$, $N_c = 1024$, $K_g = 16$, $N_g = 128$, $K_d = 16$, $N_d = 256$, $\gamma = 10$, and $\beta = 20$. More details are provided in the appendix.

4. Datasets

For the GeNeVA task, we require a dataset that contains textual instructions describing drawing actions, along with corresponding ground truth images for each instruction. To the best of our knowledge, the only such dataset publicly available is CoDraw. Additionally, we create a new dataset called i-CLEVR, specifically designed for this task.

4.1. CoDraw

CoDraw [8] is a recently released clip art-like dataset. It consists of scenes, which are sequences of images of children playing in a park. The children have different poses and expressions and the scenes include other objects such as trees, tables, and animals. There are 58 object types in total. Corresponding to every scene, there is a conversation between a Teller and a Drawer (both Amazon Mechanical Turk workers) in natural language. The Drawer updates the canvas based on the Teller's instructions. The Drawer can ask questions as well for clarification. The dataset consists of 9,993 scenes of varying length. An example of such a scene is shown in Figure 2. The initial drawing canvas x_0 for CoDraw provided to the Drawer consists of the background having just the sky and grass.

Pre-processing: In some instances of the original dataset, the Drawer waited for multiple Teller turns before modifying the image. In these cases, we concatenate consecutive turns into a single turn until the Drawer modifies the image. We also concatenate turns until a new object has been added or removed. Thus every turn has an image in which the number of objects has changed since the last turn.

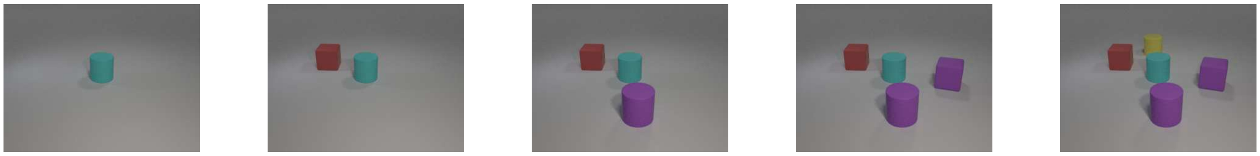
We treat the concatenated utterances of the Drawer and the Teller at time step t as the instruction, injecting a special delimiting token between the Teller and Drawer. The Teller and Drawer text contains several spelling mistakes and we run the Bing Spell Check API[1] over the entire dataset to make corrections. For words that are not present in the GloVe vocabulary, we use the "unk" word embedding from GloVe. We use the same train-valid-test split proposed in the original CoDraw dataset.

4.2. i-CLEVR

CLEVR [9] is a programmatically generated dataset that is popular in the Visual Question Answering (VQA) community. CLEVR consists of images of collections of objects with different shapes, colors, materials and sizes. Each image is assigned complex questions about object counts, attributes or existence. We build on top of the open-source generation code[2] for CLEVR to create Iterative CLEVR (i-CLEVR). Each example in the dataset consists of a sequence of 5 (image, instruction) pairs. Starting from an empty canvas (background), each instruction describes an object to add the canvas in terms of its shape and color. The instruction also describes where the object should be placed relative to existing objects in the scene. To make the task more complex and force the model to make use of context, we refer to the most recently added object by "it" instead of stating its attributes. An example from the i-CLEVR dataset

[1] https://azure.microsoft.com/en-us/services/cognitive-services/spell-check/

[2] https://github.com/facebookresearch/clevr-dataset-gen



Add a cyan cylinder at the center

Add a red cube behind it on the left

Add a purple cylinder in front of it on the right and in front of the cyan cylinder

Add a purple cube behind it on the right and in front of the red cube on the right

Add a yellow cylinder behind the purple cylinder on the left and behind the red cube on the right

Figure 4. Example sequence of image-instruction pairs from the i-CLEVR dataset.

is presented in Figure 4. The initial drawing canvas x_0 for i-CLEVR consists of the empty background. A model is tasked with learning how to add the object with the correct attributes in a plausible position, based on the textual instruction. More details about the dataset generation can be found in the appendix.

The i-CLEVR dataset consists of 10,000 sequences, totalling 50,000 images and instructions. The training split contains 6,000 sequences, while the validation and testing splits have 2,000 sequences each.

5. Experiments

In this section, we first define our evaluation metrics, and then describe the experiments carried out on the CoDraw and i-CLEVR datasets.

5.1. Evaluation Metrics

Standard metrics used for evaluating GAN models such as the Inception Score or Fréchet Inception Distance (FID) only capture how realistic the generations look relative to real images. They cannot detect if the model is correctly modifying the images according to the GeNeVA task instructions. A good evaluation metric for this task needs to identify if all the objects that were described by the Teller are present in the generated images. It should also check that the objects' positions and relationships match the instructions. To capture all of these constraints, we train an object localizer on the training dataset. For every example, we compare the detections of this localizer on the real images and the generated ones. We present the precision, recall, and F1-score for this object detection task. We also construct a graph where the nodes are objects present in the images and edges are positional relationships: left, right, behind, front. We compare the graphs constructed from the real and the generated images to test the correct placement of objects, without requiring the model to draw the objects in the same exact locations (which would have defied its generative nature).

The **object detector and localizer** is based on the Inception-v3 architecture. We modify the last layer for object detection and replace it with two heads. The first head is a linear layer with a sigmoid activation function to serve as the object detector. It is trained with a binary cross-entropy loss. The second head is a linear layer where we regress all the objects' coordinates. This head is trained with an L_2-loss with a mask applied to only compute loss over objects that occur in the ground truth image provided in the dataset. We initialize the model using pre-trained weights trained over the ILSVRC12 (ImageNet) dataset and fine-tune on the CoDraw or i-CLEVR datasets. Its performance is reported in the appendix.

Relational Similarity: To compare the arrangement of objects qualitatively, we use the above object detector/localizer to determine the type and position of objects in the ground truth and the generated image. We estimate a scene graph for each image, in which the detected objects and the image center are the vertices. The directed edges are given by the left-right and front-back relations between the vertices. To compute a relational similarity metric on scene graphs, we determine how many of the ground truth relations are present in the generated image:

$$\mathrm{rsim}(E_{G_{\mathrm{gt}}}, E_{G_{\mathrm{gen}}}) = \text{recall} \times \frac{|E_{G_{\mathrm{gen}}} \cap E_{G_{\mathrm{gt}}}|}{|E_{G_{\mathrm{gt}}}|} \qquad (6)$$

where "recall" is the recall over objects detected in the generated image w.r.t objects detected in the ground truth image. $E_{G_{\mathrm{gt}}}$ is the set of relational edges for the ground truth image corresponding to vertices common to both ground truth images and generated images, and $E_{G_{\mathrm{gen}}}$ is the set of relational edges for the generated image corresponding to vertices common to both ground truth images and generated images. The graph similarity for the complete dataset is reported by taking the mean of the final time-step value for each example over the entire dataset. This metric is a lower bound on the actual relational accuracy, as it penalizes relations based on how the objects are positioned in the ground truth image. The same instructions may, however, permit different relationship graphs. We present some examples of low-scoring to high-scoring images on this metric as well as additional discussion on rsim in the appendix.

5.2. Ablation Study

We experimented with different variations of our architecture to test the effect of each component. We define the different instantiations of our architecture as follows:

- **Baseline** The simplest version of our model. The discriminator loss only includes the adversarial terms

Model	CoDraw Precision	CoDraw Recall	CoDraw F1-Score	CoDraw $\text{rsim}(E_{G_{\text{gt}}}, E_{G_{\text{gen}}})$	i-CLEVR Precision	i-CLEVR Recall	i-CLEVR F1-Score	i-CLEVR $\text{rsim}(E_{G_{\text{gt}}}, E_{G_{\text{gen}}})$
Non-iterative	50.60	43.42	44.96	22.33	25.49	20.95	22.63	11.52
Baseline	55.61	42.31	48.05	25.31	69.09	56.38	62.08	45.19
Mismatch	62.47	48.95	54.89	32.74	71.15	60.57	65.44	50.21
G prior	60.78	49.37	54.48	33.60	82.80	77.22	79.91	63.93
Aux	54.78	51.51	53.10	33.83	83.63	75.63	79.43	55.36
D Concat	66.38	51.27	57.85	33.57	88.47	83.35	85.83	70.22
D Subtract	**66.64**	**52.66**	**58.83**	**35.41**	**92.39**	**84.72**	**88.39**	**74.02**

Table 1. Results of the GeNeVA-GAN ablation study on the CoDraw and i-CLEVR datasets.

Model	$L_{D_{\text{wrong}}}$	$f_{G_{t-1}}$	L_{aux}	D Fusion concat	D Fusion subtract
Baseline	✗	✗	✗	✗	✗
Mismatch	✓	✗	✗	✗	✗
G prior	✓	✓	✗	✗	✗
Aux	✓	✓	✓	✗	✗
D Concat	✓	✓	✓	✓	✗
D Subtract	✓	✓	✓	✗	✓

Table 2. Description of the components present in each model we test in the ablation study.

L_{fake} and L_{real}. The generator is only conditioned using the context-aware condition: $\widetilde{x}_t = G(z, h_t)$. As for the discriminator, it has no access to the previous time-step image features. Only $\tilde{x}_t$ is encoded using E_D and then passed to the discriminator D without any fusion operations.

- **Mismatch** The L_{wrong} term is added to the discriminator loss. The rest of the model is similar to the baseline.
- ***G* prior** We condition the generator on the context-free condition $f_{G_{t-1}}$ in addition to h_t as in equation (1).
- **Aux** In this model we add the L_{aux} term to both the generator and discriminator losses. The loss functions for this model follow equations (3) and (4).
- ***D* Concat** In this model, the discriminator's input is the fused features from x_{t-1} and x_t (or $\tilde{x}_t$) encoded using E_D. The fusion is a simple concatenation across the channels dimension.
- ***D* Subtract** This is the same as "*D* Concat" except for the fusion operation, which is an element-wise subtraction between the feature maps.
- **Non-iterative** The non-iterative baseline uses the same model as the "Mismatch" baseline. All the input instructions are concatenated into one instruction and the final image is generated in a single-step.

A summary of the components that are present for each model we test in the ablation study is provided in Table 2.

5.3. Results

Quantitative Results: We present the results of the ablation study in Table 1. As expected, among the iterative models, the *Baseline* model has the weakest performance on all the metrics for both datasets. This is due to the fact that it needs to construct a completely new image from scratch every time-step; there is no consistency enforced between successive generations. As for the *Mismatch* model, despite suffering from the same problems as the *Baseline*, training D to differentiate between wrong and right (image, instruction) pairs leads to generated images that better match the instructions. This is clear in Table 1 as the performance improves on all metrics compared to the *Baseline*.

The *G prior* model tries to enforce consistency between generations by using the context-free condition $f_{G_{t-1}}$. Adding this condition leads to a significant improvement to all the metrics for the i-CLEVR dataset. However, for the CoDraw dataset, it shows a less significant improvement to recall and relational similarity, while precision degrades. These results can be explained by the fact that i-CLEVR has much more complex relationships between objects and the instructions have a strong dependence on the existing objects in the scene. Therefore, the model benefits from having access to how the objects were placed in the most recent iteration. As for CoDraw, the relationships among objects are relatively simpler. Nevertheless, adding the context-aware condition helps with placing the objects correctly as shown by the improvement in the relational similarity metric. A possible drawback from using the context-free condition is that it is harder to recover from past mistakes, especially if it has to do with a large objects. This drawback can explain the drop in precision.

For the *Aux* model, it had different effects on the two datasets. For CoDraw, it helped improve recall and relational similarity, but caused a significant decrease in precision. For i-CLEVR, it helped improve precision with hurting the recall and relational similarity. This different behavior for each dataset can be explained by the types of objects that are present. While for CoDraw, there are objects that are almost always present like the girl or the boy, for i-CLEVR there is high randomness in objects presence. Adding the auxiliary objective encourages the model to make sure the frequent objects are present, leading to the increase in recall while hurting precision. Finally, we observe that giving D access to the previous image x_{t-1} shows im-

Drawer: ready
Teller: large apple tree left side trunk start 2 3 way up green and about 1 and 1 4 inches from left side

Teller: big cloud right side almost touching apple tree 1 2 inch up into blue

Teller: below the cloud full size girl her head touches top of green hands over head centered under cloud

Teller: boy 1 1 2 to left of girl under right side of tree same height as girl facing right hands out to right holding a football in both

Teller: sandbox medium size lower left corner facing right left side off screen lower right corner is equal to end of tree trunk
Drawer: yes

Add a blue cube at the center

Add a red cylinder behind it on the left

Add a purple sphere in front of it on the right and in front of the blue cube

Add a cyan cylinder behind it on the right and in front of the red cylinder on the right

Add a yellow sphere in front of the red cylinder on the left and on the left of the blue cube

Figure 5. Example images generated by our model (D Subtract) on CoDraw (top row) and i-CLEVR (bottom row); shown with the provided instructions. We scale images from both datasets to 128x128 in a pre-processing step.

provement on almost all the metrics for both datasets. We also observe that subtraction fusion consistently performs better than concatenation fusion and outperforms all other models for both datasets. This indicates that encouraging the discriminator to focus on the modifications gives a better training signal for the generator.

The *Non-iterative* model performs worse than all of the iterative models. This is likely because the language encoder has difficulty understanding dependencies and object relationships in a lengthy concatenated instruction. The benefit of using an iterative model is more visible in the i-CLEVR dataset since in it, the spatial relationships are always defined in terms of existing objects. This makes it very difficult to comprehend all the relationships across different turns in a single step. By having multiple steps, iterative generation makes this task easier. The results of this experiment make a case for iterative generation in complex text-conditional image generation tasks that have traditionally been performed non-iteratively.

Qualitative Results: We present some examples of images generated by our model in Figure 5. Due to space constraints, more example images are provided in the appendix. On CoDraw, we observe that the model is able to generate scenes consistent with the conversation and generation history and gets most of the coarse details correct, such as large objects and their relative positions. But it has difficulty in capturing fine-grained details, such as tiny objects, facial expressions, and object poses. The model also struggles when a single instruction asks to add several objects at once. For i-CLEVR, the model captures spatial relationships and colors very accurately as demonstrated in Figure 5. However, in some instances, the model fails to add the fifth object when the image is already crowded and there is no space left to add it without moving the others. We also experimented with using an intermediate ground truth image as the initial image at test time and the model was able to generalize and place objects correctly in that scenario as well. The results of this experiment are presented in the appendix.

6. Conclusion and Future Work

We presented a recurrent GAN model for the GeNeVA task and show that the model is able to draw reasonable images for the provided instructions iteratively. It also significantly outperforms the non-iterative baseline. We presented an ablation study to highlight the contribution of different components. Since this task can have several plausible solutions and no existing metric can capture all of them, we proposed a relational similarity metric to capture the possible relationships. For future research directions, having a system that can also ask questions from the user when it needs clarifications would potentially be even more useful. Collecting photo-realistic images, transitions between such images, and annotations in the form of instructions for these transitions is prohibitively expensive; hence, no photo-realistic dataset appropriate for this task publicly exists. Such datasets are needed to scale this task to photo-realistic images.

Acknowledgements

We thank Philip Bachman for valuable discussions.

References

[1] Mehdi Mirza and Simon Osindero, "Conditional generative adversarial nets," arXiv:1411.1784 [cs.AI], 2014.

[2] Shikhar Sharma, Dendi Suhubdy, Vincent Michalski, Samira Ebrahimi Kahou, and Yoshua Bengio, "Chatpainter: Improving text to image generation using dialogue," in *International Conference on Learning Representations (ICLR) Workshop*, 2018.

[3] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee, "Generative adversarial text to image synthesis," in *International Conference on Machine Learning (ICML)*, 2016.

[4] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas, "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *International Conference on Computer Vision (ICCV)*, 2017.

[5] Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee, "Inferring semantic layout for hierarchical text-to-image synthesis," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[6] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He, "AttnGAN: Fine-grained text to image generation with attentional generative adversarial networks," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick, "Microsoft COCO: Common objects in context," in *European Conference on Computer Vision (ECCV)*, 2014.

[8] Jin-Hwa Kim, Nikita Kitaev, Xinlei Chen, Marcus Rohrbach, Byoung-Tak Zhang, Yuandong Tian, Dhruv Batra, and Devi Parikh, "CoDraw: Collaborative drawing as a testbed for grounded goal-driven communication," in *Proceedings of the 57th Conference of the Association for Computational Linguistics*, 2019, pp. 6495–6513. [Online]. Available: https://www.aclweb.org/anthology/P19-1651

[9] Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick, "Inferring and executing programs for visual reasoning," in *International Conference on Computer Vision (ICCV)*, 2017.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems 27*, 2014.

[11] Augustus Odena, Christopher Olah, and Jonathon Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *International Conference on Machine Learning (ICML)*, 2017.

[12] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M.F. Moura, Devi Parikh, and Dhruv Batra, "Visual Dialog," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[13] Jianwei Yang, Anitha Kannan, Dhruv Batra, and Devi Parikh, "LR-GAN: Layered recursive generative adversarial networks for image generation," in *International Conference on Learning Representations (ICLR)*, 2017.

[14] Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, Eli Shechtman, and Simon Lucey, "ST-GAN: Spatial transformer generative adversarial networks for image compositing," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[15] Jeffrey Pennington, Richard Socher, and Christopher Manning, "Glove: Global vectors for word representation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014.

[16] Takeru Miyato and Masanori Koyama, "cGANs with projection discriminator," in *International Conference on Learning Representations (ICLR)*, 2018.

[17] Jae Hyun Lim and Jong Chul Ye, "Geometric GAN," arXiv:1705.02894 [stat.ML], 2017.

[18] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[19] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena, "Self-attention generative adversarial networks," arXiv:1805.08318 [stat.ML], 2018.

[20] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin, "Which training methods for GANs do actually converge?" in *International Conference on Machine learning (ICML)*, 2018.

[21] Jimmy Ba, Ryan Kiros, and Geoffrey E. Hinton, "Layer normalization," arXiv:1607.06450 [stat.ML], 2016.

[22] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *Journal of Machine Learning Research (JMLR)*, 2015.

[23] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization." in *International Conference on Learning Representations (ICLR)*, 2015.