SSCR: Iterative Language-Based Image Editing via Self-Supervised Counterfactual Reasoning

Tsu-Jui Fu[†], Xin Eric Wang[‡], Scott T. Grafton[†], Miguel P. Eckstein[†], William Yang Wang[†]

[†]UC Santa Barbara [‡]UC Santa Cruz

tsu-juifu@ucsb.edu, {scott.grafton, miguel.eckstein}@psych.ucsb.edu

william@cs.ucsb.edu, xwang366@ucsc.edu

Abstract

Iterative Language-Based Image Editing (ILBIE) tasks follow iterative instructions to edit images step by step. Data scarcity is a significant issue for ILBIE as it is challenging to collect large-scale examples of images before and after instruction-based changes. However, humans still accomplish these editing tasks even when presented with an unfamiliar image-instruction pair. Such ability results from counterfactual thinking and the ability to think about alternatives to events that have happened already. In this paper, we introduce a Self-Supervised Counterfactual Reasoning (SSCR) framework that incorporates counterfactual thinking to overcome data scarcity. SSCR allows the model to consider out-of-distribution instructions paired with previous images. With the help of cross-task consistency (CTC), we train these counterfactual instructions in a self-supervised scenario. Extensive results show that SSCR improves the correctness of ILBIE in terms of both object identity and position, establishing a new state of the art (SOTA) on two IBLIE datasets (i-CLEVR and CoDraw). Even with only 50% of the training data, SSCR achieves a comparable result to using complete data.

1 Introduction

Digital design tools like Illustrator or Photoshop are widely used nowadays. Though having considerable user demand, they require prior knowledge and multiple steps to use successfully. These applications would significantly improve the accessibility if they can automatically perform corresponding editing actions based on the language instructions given by users for each step.

Iterative language-based image editing (ILBIE) task follows iterative instructions to edit images step by step, as illustrated in Fig. 1. To accomplish ILBIE, models are required not only to modify images but also to understand the visual differences between the previous and resulting image, based on the given instructions. One of the primary limitations of ILBIE is data scarcity. Since collecting large-scale previous-resulting images with instructions is difficult, it makes learning the association between vision and language challenging.

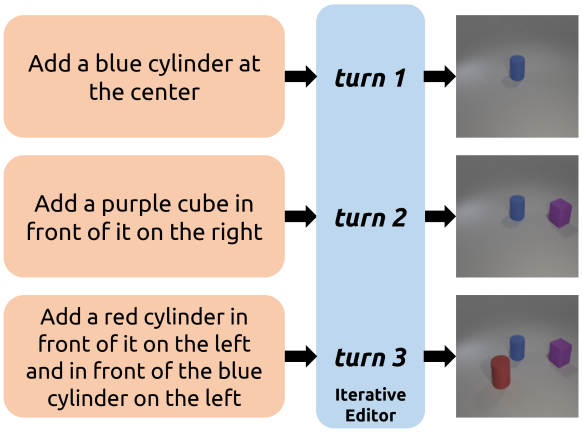

Figure 1: An example of the iterative language-based image editing (ILBIE) task. During each turn, the model edits the image from the previous turn based on the current instruction. Eventually, a desired image is accomplished after iterative editing. Note that the generation is at the pixel level.

A GAN-based (Goodfellow et al., 2015) model, GeNeVA (El-Nouby et al., 2019), is proposed to perform ILBIE, where a conditional generator serves as the image editor, and a discriminator provides the training loss by discriminating a resulting image. Though it yields promising results, GeNeVA neglects the data scarcity issue of ILBIE. As a binary classifier, the discriminator easily suffers from data scarcity given the shortage of ground-truth instruction-and-resulting-image pairs and thus limits the model's generalization ability to new instructions.

Despite lacking prior experiences with images or instructions, humans can still accomplish editing under unfamiliar image-instruction pairs. For ex-

ample, for a given instruction "add a purple cube in front of the blue cylinder", humans can think about the resulting image if the instruction changed to "adding a blue square" or "on the right of". This process is known as counterfactual thinking (Roese, 1997), which allows humans to operate in data-scarce scenarios by considering alternative instructions from the seen examples.

In this paper, we introduce self-supervised counterfactual reasoning (SSCR) that incorporates counterfactual thinking to deal with the data scarcity issue. SSCR allows the model to think about expected, resulting images under unseen instructions. Since there are no ground-truth resulting images, we propose cross-task consistency (CTC), which adopts an iterative explainer to reconstruct the instruction of each step. With CTC, we can supply detailed token-level training loss (e.g., wrong objects or wrong positions), which is better than only using the discriminator, and consider these counterfactual instructions in a self-supervised scenario.

The experimental results on i-CLEVR (El-Nouby et al., 2019) and CoDraw (Kim et al., 2019) show that our SSCR can improve the correctness of the ILBIE task in both aspects of object identity and position. In summary, our contributions are three-fold:

- We introduce SSCR that incorporates counterfactual thinking into the ILBIE task to deal with the data scarcity issue.
- The proposed cross-task consistency (CTC) and counterfactual reasoning methods help train the generator better, improve the generalizability, and achieve the SOTA results on i-CLEVR and CoDraw.
- Extensive ablation studies show that SSCR is effective even with only partial training data.

2 Related Work

Text-to-Image (T2I) generates an image that matches the given instruction. T2I is challenging yet important that has a vast potential in practical applications like art generation or automatic design (Nguyen et al., 2017; Reed et al., 2017; Tan et al., 2019). With the success of a generative adversarial network (Goodfellow et al., 2015) on the image generation task, several works (Reed et al., 2016; Zhang et al., 2017; Xu et al., 2018) introduce different GAN-based models to synthesize an image from a text description. Unlike T2I, we focus on image editing, where a model needs to understand the visual differences between two images rather than generating an image from scratch.

Language-based Image Editing (LBIE) tasks a model to edit an image based on the guided text description. PixelTone (Laput et al., 2013) and Image Spirit (Cheng et al., 2013) are both rule-based, which accept only pre-defined instructions and semantic labels that can significantly decrease the practicality of LBIE. Some studies (Chen et al., 2018; Shinagawa et al., 2017) adopt the conditional GAN model to attend on the instruction and perform LBIE as image colorization. However, image colorization is not truly an editing task since it only supports fixed object templates, and the objects and the scene of the image remain the same after editing. In contrast, the editing processes of Photoshop or Illustrator are not accomplished in a single pass. GeNeVA (El-Nouby et al., 2019) proposes an iterative GAN-based generator to accomplish iterative language-based image editing (ILBIE) but neglects the data scarcity issue.

Counterfactual Thinking (Roese, 1997) is a concept that refers to the human propensity to consider possible alternatives to events that have happened already. People can consider different outcomes from a wide range of conditions and engage in causal reasoning by asking questions like "What if ...?" or "If I had only...." Previous works (Kusner et al., 2017; Garg et al., 2019) have shown how counterfactual fairness improves the robustness of the model and makes it more explainable. Furthermore, counterfactual thinking has also been applied to augment training targets (Zmigrod et al., 2019; Fu et al., 2020). In this paper, we incorporate counterfactual thinking into that ILBIE task that considers counterfactual instructions to deal with the data scarcity issue and improve the generalizability.

3 Self-Supervised Counterfactual Reasoning (SSCR)

3.1 Task Definition

During each turn t, an editor edits the image from the previous turn V_{t-1} into the current turn V_t based on instruction I_t. After a final turn T, we get the predicted final image V_T and evaluate the outcome with the ground truth resulting image O_T. Note that the editing process is at a pixel level where the model has to generate each pixel of the image:

$$\begin{aligned} V_t &= \text{Editor}(V_{t-1}, I_t), \\ \text{eval} &= \text{Compare}(V_T, O_T). \end{aligned} \tag{1}$$

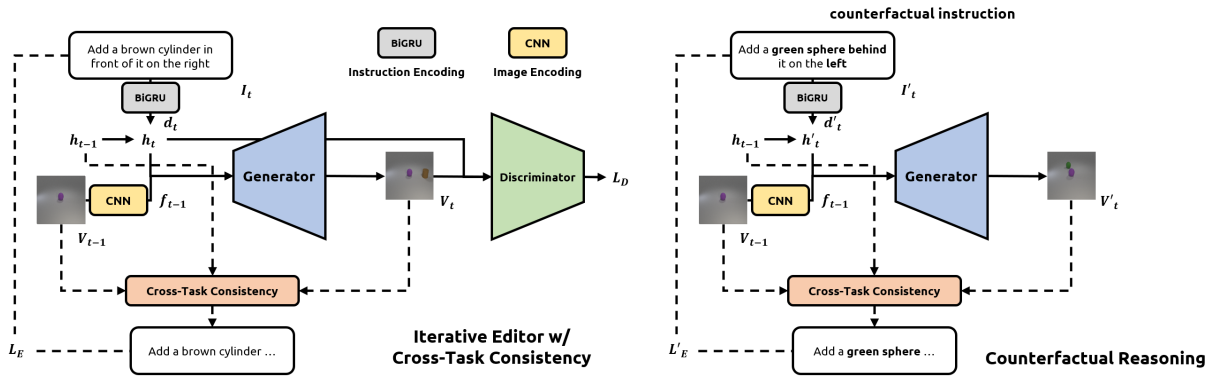

Figure 2: An overview of our self-supervised counterfactual reasoning (SSCR). The iterative editor modifies an image based on current instruction and editing history. Counterfactual reasoning allows the model to think about various counterfactual instructions that can improve the generalizability and deal with data scarcity. Since there are no ground-truth images, we propose cross-task consistency (CTC) to not only provide explicit training signal but also train these counterfactual instructions self-supervisedly.

3.2 Overview

To overcome data scarcity, we introduce self-supervised counterfactual reasoning (SSCR). The overall framework is illustrated in Fig, 2. The iterative editor is a conditional generator that modifies an image based on current instruction and editing history.

Counterfactual reasoning allows the model to think about the expected, resulting images under various counterfactual instructions. In this way, the editor can consider more diverse instructions than the original data to improve the generalizability, even if under data scarcity. Since there are no ground-truth resulting images for these counterfactual instructions, we propose cross-task consistency (CTC). CTC adopts an iterative explainer to reconstruct the given instruction of each editing step. With the help of this cross-task matching, we can not only provide a detailed token-level training signal to train the editor better but also supply training loss for counterfactual reasoning in a self-supervised scenario.

3.3 Iterative Editor

Similar to GeNeVA (El-Nouby et al., 2019), the iterative editor is a GAN-based architecture that contains a conditional generator G and a discriminator D. We first apply a bidirectional GRU (Chung et al., 2014) to encode the instruction I_t as d_t for each turn t. And another GRU is used to encode the history of instructions h_t as following:

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

Then, to perform the editing for turn t, we adopt a convolutional neural network (CNN) (Miyato and Koyama, 2018) to extract image features f_{t-1} from the previous image V_{t-1}, concatenate with the instruction history h_t, and feed into G to predict the resulting image V_t:

$$V_t = G([f_{t-1}, h_t]). \tag{3}$$

After all iterations, there is the final image V_T after the final turn T.

For each turn, D provides a binary training signal by discriminating a resulting image that is generated from either G or the ground-truth data according to the instruction history h_t:

$$L_G = \sum_{t=1}^{T} \mathbb{E}_{V_t \sim \mathcal{P}_{G_t}}[\log(D([V_t, h_t]))], \tag{4}$$

where L_G is the binary loss from D.

For training D, similar to T2I (Reed et al., 2016), we add additional [real image, wrong instruction] pairs as false examples:

$$L_D = \sum_{t=1}^{T} L_{D_{\text{real}_t}} + \frac{1}{2}(L_{D_{\text{false}_t}} + L_{D_{\text{wrong}_t}}), \tag{5}$$

where

$$\begin{aligned} L_{D_{\text{real}_t}} &= \mathbb{E}_{O_t \sim \mathcal{P}_{\text{data}}}[\log(D([O_t, h_t]))], \\ L_{D_{\text{false}_t}} &= \mathbb{E}_{V_t \sim \mathcal{P}_{G_t}}[\log(1 - D([V_t, h_t]))], \\ L_{D_{\text{wrong}_t}} &= \mathbb{E}_{O_t \sim \mathcal{P}_{\text{data}}}[\log(1 - D([O_t, h'_t]))], \end{aligned} \tag{6}$$

with ground-truth data distribution $\mathcal{P}_{\text{data}}$ and h'_t being the wrong instruction history by randomly selecting another instruction.

Then G and D are optimized through an alternating minmax game:

$$\max_G \min_D L_G + L_D. \tag{7}$$

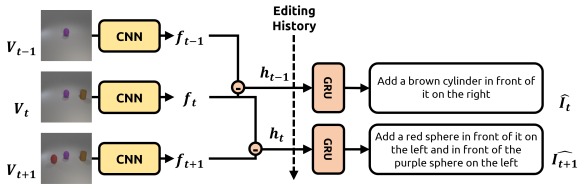

Figure 3: The architecture of our iterative explainer. We consider the previous-resulting image pair and the encoded instruction history as input to reconstruct the editing instruction by an attention-based GRU decoder.

3.4 Cross-Task Consistency (CTC)

Though we can train the iterative editor for ILBIE, D only supports a binary training loss, which is not explicit enough to express the complex association between the visual difference and the text description. To supply a more explicit training loss, we propose cross-task consistency (CTC). Despite being image generation, we consider instruction generation, which explains the visual difference between previous-resulting image pairs, to do reasoning for the editing process in a cross-task scenario. During CTC, an iterative explainer provides a token-level training signal that encourages the matching between the predicted image and the original instruction.

Iterative Explainer Our iterative exaplainer E is an instruction decoder which considers previous-resulting image pair and the instruction history as input, as shown in Fig. 3:

$$\hat{I}_t = E(V_t, V_{t-1}, h_{t-1}). \tag{8}$$

Similar to the iterative editor, we apply CNN to extract visual feature f for both previous and predicted resulting image:

$$f_{t-1} = \text{CNN}(V_{t-1}), f_t = \text{CNN}(V_t). \tag{9}$$

Then, a GRU serves as an attention-based language decoder (Xu et al., 2015) which reconstructs the instruction $\hat{I}_t$ according to the feature difference and instruction history h_{t-1} of previous turn:

$$\begin{aligned} g_0 &= [f_d, h_{t-1}], \\ \hat{w}_i, g_i &= \text{GRU}(w_{i-1}, g_{i-1}), \\ \hat{I}_t &= \{\hat{w}_1, \hat{w}_2, ..., \hat{w}_L\}, \end{aligned} \tag{10}$$

where $f_d = f_t - f_{t-1}$ represents the visual difference by subtracting previous and result feature, g_i is the decoding history, and $\hat{w}_i$ is the predicted word token of the instruction. All w_i are combined as the reconstruction where L is the length of the instruction. The iterative explainer considers not only the visual difference but also instruction history so that we can reconstruct the instruction, which explains the editing of the resulting image following by the editing history.

Dataset	Token Type	Example
i-CLEVR	color	blue, purple
	object	cylinder, cube
	relation	at the center, in front of
CoDraw	size	small, meidum
	object	sun, boy
	relation	in the middle, on the left

Table 1: Types of token on i-CLEVR and CoDraw.

Finally, we provide an explicit token-level training signal L_E by computing the teacher-forcing loss (Williams and Zipser, 1989) between the original instruction I_t and the reconstructed one $\hat{I}_t$:

$$L_E = \sum_{i=1}^{L} \text{CELoss}(\hat{w}_i, w_i), \tag{11}$$

where w_i is the ith token of I_t and CELoss means the cross-entropy loss.

By minimizing L_E, G learns to match the original instruction with this cross-task consistency. Different from L_G, which only supplies binary but vague loss, L_E provides token-level loss about the information of the wrong object or wrong position (by comparing $\hat{w}_i$ with w_i) that can train G better for each editing turn. In the experiments, E is pre-trained by the ground-truth image pairs and is fixed during the following training.

3.5 Counterfactual Reasoning

We assume that $\mathcal{U}$ is the available training data. Because of the practical challenge of collecting large-scale previous-resulting images with instructions, $\mathcal{U}$ suffers from data scarcity. To deal with this issue, we propose counterfactual reasoning to allow the model to consider various instructions out of the distribution of $\mathcal{U}$. For instance, an instruction $I' \sim \mathcal{U}'$ from the intervention data $\mathcal{U}'$ replaces the original instruction, and we edit the image based on the counterfactual instruction I'.

Instruction Intervention To get the intervention data U' that provides diverse instructions, we do interventions on the original instructions I:

$$\begin{aligned} I, O &= \mathcal{U}, \\ I' &= \text{intervention}(I), \\ \mathcal{U}' &= \{I', O\}, \end{aligned} \tag{12}$$

where O is the image in the original $\mathcal{U}$.

Algorithm 1 Iterative Editor with CTC

G: the generator model
D: the discriminator model
H: the instruction history encoder
E: our iterative explainer in CTC
$\mathcal{U}$: the original training set

Pre-train E with $\mathcal{U}$
Initialize G, D

while TRAIN_EDITOR **do**
  for $t \leftarrow 1$ to T **do**
    $I_t, O_t \leftarrow \mathcal{U}$
    $h_t \leftarrow H(h_{t-1}, I_t)$
    $V_t \leftarrow G(h_t, O_{t-1})$ ▷ teacher-forcing training
    $\hat{I}_t \leftarrow E(V_t, O_{t-1}, h_{t-1})$

    $L_G \leftarrow$ binary loss from D ▷ Eq. 4
    $L_E \leftarrow$ explicit loss from E ▷ Eq. 11
    Update G by maximizing $L_G - L_E$

    $L_D \leftarrow$ loss for D ▷ Eq. 5
    Update D by minimizing L_D
  end for
end while

First, we apply NLTK (Bird and Loper, 2004) to parse out *tokens* in the original I. The types of *token* on i-CLEVR and CoDraw are shown in Table 1.

We then replace these *tokens* with randomly sampled *tokens* of the same type to get the counterfactual I'. Finally, I' combines with the original image G as the intervention data $\mathcal{U}'$. Our experiments show that this simple yet effective intervention makes the training data more diverse and deals with data scarcity during our counterfactual reasoning.

For each turn t, with I'_t from U', we predict the counterfactual resulting image V'_t:

$$V'_t = G([f_{t-1}, h'_t]), \tag{13}$$

where h'_t is the counterfactual instruction history encoded from I'.

Since there is no ground-truth image for the counterfactual instruction I'_t, we adopt the iterative explainer E to provide counterfactual training loss L'_E in a self-supervised scenario:

$$\begin{aligned} \hat{I}'_t &= E(V'_t, V_{t-1}, h_{t-1}), \\ L'_E &= \sum_{i=1}^{L} \text{CELoss}(\hat{w'_i}, w'_i), \end{aligned} \tag{14}$$

where $\hat{w'_i}$ and w'_i are the ith word token.

By minimizing L'_E, the model has an opportunity to access U', which is different from the original training data. With the help of our iterative explainer, SSCR improves the generalizability by reasoning diverse counterfactual instructions I' even if under data scarcity.

Algorithm 2 Counterfactual Reasoning

while COUNTERFACTUAL_REASONING **do**
  for $t \leftarrow 1$ to T **do**
    $\mathcal{U}' \leftarrow$ intervention($\mathcal{U}$)
    $\_, O_t \leftarrow \mathcal{U}$
    $I'_t, \_ \leftarrow \mathcal{U}'$ ▷ counterfactual instruction

    $h_t \leftarrow H(h_{t-1}, I_t)$ ▷ real history
    $h'_t \leftarrow H(h_{t-1}, I'_t)$ ▷ counterfactual history
    $V'_t \leftarrow G(h'_t, O_{t-1})$ ▷ counterfactual editing
    $\hat{I}'_t \leftarrow E(V'_t, O_{t-1}, h_{t-1})$

    L'_E = counterfactual loss from E ▷ Eq. 14
    Update G by minimizing L'_E
  end for
end while

3.6 Learning of SSCR

Alg. 1 presents the learning process of training the iterative editor with CTC. Since ILBIE is also a sequential generation process, we apply the widely used teacher-forcing where we feed in the ground-truth resulting image (O_{t-1}) from the previous turn instead of our predicted one (V_{t-1}) to make the training more robust. When training the iterative editor, for each turn t, we adopt G to perform image editing. We maximize the binary loss from D (L_G) with minimizing the explicit token-level loss from E (L_E) to train G. We also update D by minimize L_D:

$$\max_G \min_D L_G + L_D - L_E. \tag{15}$$

During counterfactual reasoning, as shown in Alg. 2, we first perform an intervention on $\mathcal{U}$ to get the counterfactual instructions (I'). Then, we edit the image based on I'. Since there is no ground-truth resulting image for the counterfactual editing, we adopt CTC to compute the cycle-consistency loss (L'_E) self-supervisedly. Similar to the iterative editor, we also apply teacher-forcing training (feeding in O_{t-1} and h_{t-1}) to further update G. In this way, G can improve the generalizability by considering the counterfactual U', which is more diverse than U.

4 Experiments

4.1 Experimental Setup

Datasets We evaluate our counterfactual framework on two ILBIE datasets, i-CLEVR (El-Nouby

	i-CLEVR				CoDraw			
Method	Precision	Recall	F1	RelSim	Precision	Recall	F1	RelSim
GeNeVA	71.01	42.61	53.26	30.66	54.38	54.42	54.40	38.93
w/ CTC only	72.24	45.51	55.84	33.67	57.69	55.60	56.62	38.68
w/ SSCR	**73.75**	**46.39**	**56.96**	**34.54**	**58.17**	**56.61**	**57.38**	**39.11**

Table 2: The testing results of the baseline (GeNeVA[1]), with only cross-task consistency (CTC only), and with whole self-supervised counterfactual reasoning (SSCR) for both i-CLEVR and CoDraw.

et al., 2019) and CoDraw (Kim et al., 2019). Each example in i-CLEVR consists of a sequence of 5 (image, instruction) pairs. The instruction describes where the object should be placed relative to existing objects. In i-CLEVR, there are 6,000, 2,000, and 2,000 examples for training, validation, and testing, respectively.

CoDraw is a more difficult art-like dataset of children playing in a park. There are 58 objects and children with different poses. We use the same split as in CoDraw where 7,988, 1,002, and 1,002 are for training, validation, and testing.

Evaluation Metrics Standard metrics like Inception Score (Salimans et al., 2016) or Frechet Inception Distance (FID) (Heusel et al., 2017) cannot detect whether the editing is correct based on the instruction (El-Nouby et al., 2019). Following GeNeVA (El-Nouby et al., 2019), we adopt F1 and RelSim to evaluate the editing result.

The F1 score is based on a pre-trained object detector (Szegedy et al., 2016) (˜99% accuracy on both i-CLEVR and CoDraw), which detects the objects in the predicted images that meet the ground-truth resulting images. To evaluate not only object type but also object position, we build the scene graph according to the object detector. The edges are given by the left-right and front-back relations between the vertices (objects). Then, RelSim determines how many of the ground-truth relations are in the predicted images:

$$\text{RelSim}(E_{\text{gt}}, E_{\text{pd}}) = \text{recall} \times \frac{|E_{\text{pd}} \cap E_{\text{gt}}|}{|E_{\text{gt}}|}, \quad (16)$$

where E_{gt} and E_{pd} are relational edges for ground-truth resulting images and predicted image. Note that we only evaluate the final predicted image of each example for both F1 and RelSim.

Baseline We use the SOTA model GeNeVA[1] as our baseline: it shares the same model architecture as our iterative editor and is trained with the GAN objective but without the cross-task consistency (CTC) and our counterfactual reasoning.

Implementation Detail We apply the ResBlocks (Miyato and Koyama, 2018) into G and D where the visual feature size is 1024. For our E, we add self-attention (Zhang et al., 2019) for the concatenation of the visual difference and the encoded instruction history. We adopt Adam (Kingma and Ba, 2015) to optimize the iterative editor with learning rate 1e-4 for L_G and L_E, 4e-4 for L_D. The learning rate of L'_E during the counterfactual reasoning is 5e-5.

4.2 Quantitative Results

Table 2 presents the testing F1 and RelSim results. First, with our cross-task consistency (CTC only), which provides a more explicit training signal, we can improve the baseline on the i-CLEVR dataset in terms of all metrics. In particular, CTC improves 1.2% on precision, 2.9% on recall, and 2.6% on F1. Additionally, for whole self-supervised counterfactual reasoning (SSCR), which allows the model to consider out-of-distribution instructions, it brings more improvements and achieves new SOTA results, *e.g.*, 56.9% on F1 and 34.5% on RelSim.

Similar trends can be found on CoDraw. Since the instructions under CoDraw are more complex, the improvement of relation correctness (RelSim) is not as high as i-CLEVR. But for object correctness, CTC still improves baseline with 2.2% F1, and SSCR further achieves the new SOTA on all metrics, *e.g.*, 57.4% F1 and 39.1% RelSim.

4.3 Ablation Study

Under Data Scarcity To examine the framework's effectiveness under the data scarcity scenario, we compare models trained using 100%, 80%, and 50% data. Note that our E is also pre-trained using the same 100%, 80%, and 50% data. The results are shown in Fig. 4.

We can observe that on both i-CLEVR and CoDraw datasets, the baseline performance drops dras-

[1]We reproduce the result for GeNeVA by their official GitHub repo (https://github.com/Maluuba/GeNeVA). We apply the default hyperparameters as them, and issue #2 can support that the results are comparable.

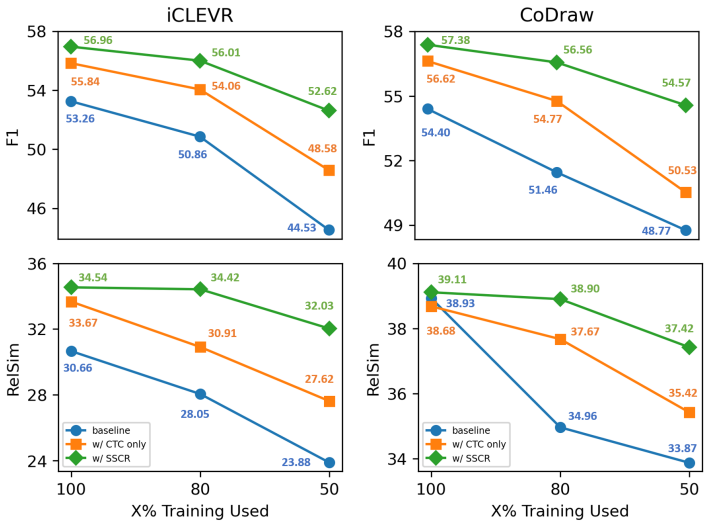

Figure 4: Result comparison among baseline, with only cross-task consistency (CTC only), and with whole self-supervised counterfactual reasoning (SSCR) under different ratios of training data. Note that the iterative explainer is also pre-trained using the same available data for each result.

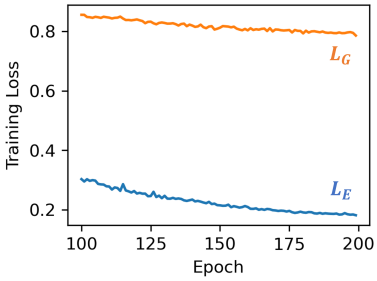

Figure 5: The learning curve of training loss provided from the discriminator (L_G) and our iterative explainer (L_E) on i-CLEVR.

X% Training Used	PPL	BLEU
100%	0.1073	50.236
80%	0.1295	48.873
50%	0.1163	48.763

Table 3: The PPL and BLEU of our iterative explainer with different ratios of training data on i-CLEVR.

tically as the training data decreases, and our SSCR consistently outperforms the baseline. More importantly, the baseline severely suffers from the data scarcity issue, while SSCR is relatively resilient to data decrease and only drops 4.34 F1 score and 2.51 RelSim score (*vs.* 8.73 and 6.78 reduced by the baseline) on iCLEVR when there is only 50% data. Similar results can be observed on CoDraw. Furthermore, comparing SSCR with 50% data and the baseline with 100%, we can notice that our method can achieve comparable results to the baseline with only half the data used for training. Therefore, incorporating counterfactual thinking to explore out-of-distribution instructions indeed makes the model better capable of generalization and avoiding performances drops from data scarcity.

Table 3 presents the performance of our iterative explainer E with different ratios of training examples. Perplexity (PPL) and BLEU (Papineni et al., 2002) are calculated between the reconstructed instructions and the original ones. We can see that the PPL and BLEU under 50% are similar to 100%. It shows that E still supplies meaningful training loss for SSCR even if only using 50% data.

Zero-shot Generalization To further demonstrate the effectiveness of SSCR under severe data scarcity, we conduct a zero-shot experiment for the i-CLEVR dataset. The zero-shot setting is as following. There are 3 shapes (*cube*, *sphere*, *cylinder*) and 8 colors (*gray*, *red*, *blue*, *green*, *brown*, *purple*, *cyan*, *yellow*), which lead to 24 different objects

	100%		50%	
Method	F1	RelSim	F1	RelSim
GeNeVA	53.26	30.66	44.53	23.88
w/ SSCR (D)	54.05	30.87	43.31	22.99
w/ SSCR (E)	**56.95**	**34.54**	**52.62**	**32.03**

Table 4: Results of discriminator (D) or iterative explainer (E) used for the counterfactual reasoning (SSCR) on i-CLEVR.

Method	F1	RelSim
GeNeVA	42.23	23.70
w/ CTC Only	<u>43.91</u>	<u>25.26</u>
w/ SSCR	**48.30**	**29.09**

Table 5: Results of zero-shot generalization.

on the i-CLEVR dataset. We remove examples containing "*gray cube*, *red cube*, *green sphere*, or *purple cylinder*" in the training set but still evaluate the full testing set with all kinds of objects.

The result is shown in Table 5. Since there is no example like "*gray cube*" in the training set, CTC can only consider those seen objects and improves marginally. However, the iterative explainer (E) can disentangle color and shape information from "*gray sphere*" and "*green cube*," and generalize to the unseen object "*gray cude*". During SSCR, when we intervene the counterfactual instructions to contain "*gray cube*," the iterative explainer can still provide self-supervised loss to make the model consider unseen objects. Hence, SSCR can bring out obvious improvements on both F1 and RelSim, even if under the zero-shot setting.

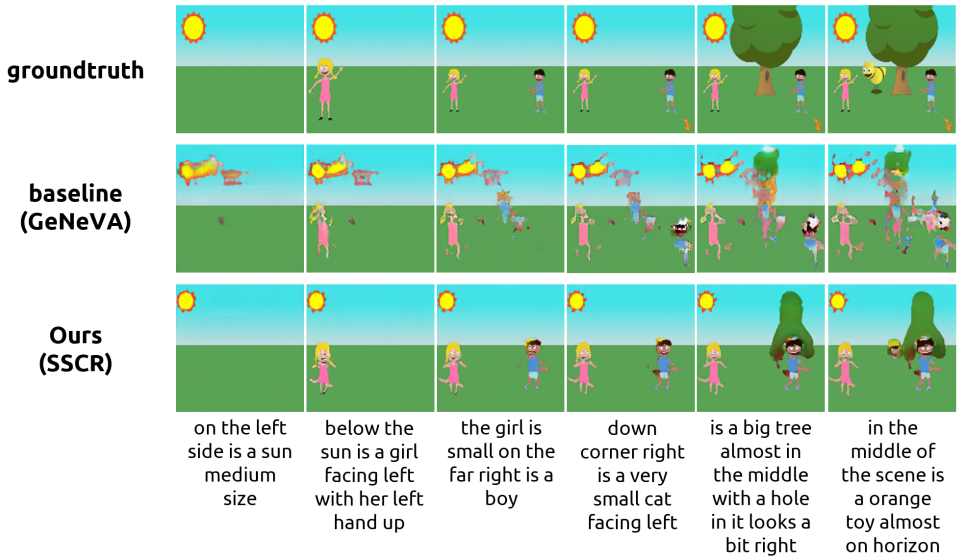

Figure 6: Visulaization example of baseline (GeNeVA) and our SSCR on CoDraw.

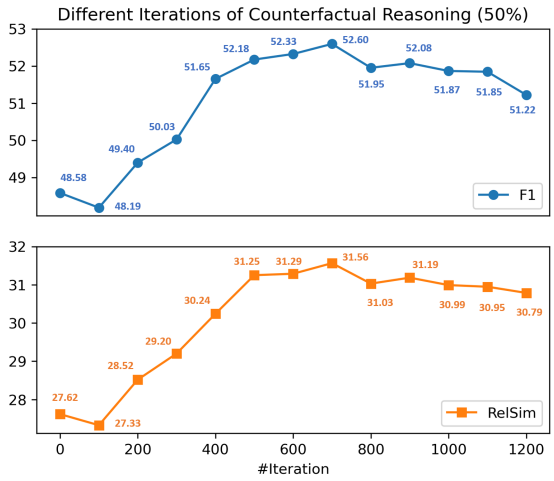

Figure 7: The validation F1 and RelSim on i-CLEVR with 50% training data under different instructions of counterfactual reasoning.

Iterative Explainer *vs*. Discriminator Fig. 5 shows the learning curve of the training losses of the discriminator D (L_G) and our iterative explainer E (L_E). We can see that the relative decrease of L_G over time is very little, which means that D can barely provide extra training signal after 100 epochs. In contrast, since E can supply explicit token-level loss instead of vague binary loss, L_E keeps decreasing much and training the model.

Table 4 shows the comparison when using D and our E to provide the training loss during the counterfactual reasoning. If using D, since there are not ground-truth resulting images of those counterfactual instructions, we cannot feed them into D as true examples. It can only provide training loss by discriminating predicted images as false. Therefore, using D during SSCR cannot improve the model much, and may even hurt the generalizability under data scarcity, *e.g.*, 23.9 drops to 23.0 on RelSim for 50%.

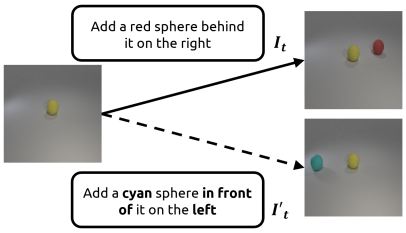

Figure 8: Example of counterfactual reasoning.

In comparison, since our E does not suffer from data scarcity, it supports SSCR by providing meaningful training loss to perform counterfactual reasoning, and thus improves the generalizability, *e.g.*, 23.9 increases to 32.0 on RelSim for 50%.

Counterfactual Reasoning: The More The Better? Despite allowing the model to explore various instructions and become more generalized, excessive counterfactual reasoning may result in overfitting to existing images and degrade the performance. Fig. 7 presents the validation performance under different iterations. It shows a trade-off between the model's generalizability and the iterations of the counterfactual reasoning. The performance keeps improving until the best 700 iteration and then drops down, possibly due to overfitting to existing images and the imperfect cost function for instruction prediction.

Qualitative Results Fig. 8 present an example of counterfactual instructions and the predicted resulting image. We replace *color token* "red" with

"cyan"", *color token* "behind" with "in front of", and "right" with "left." By considering counterfactual instructions, SSCR allows the model to explore diverse instruction-image pairs, that deals with the data scarcity issue.

Fig. 6 demonstrates an example of the iterative editing on CoDraw. For baseline GeNeVA, since there is only a discriminator to provide vague loss that the pixels of those generated objects are almost broken, it makes the predicted images low quality. In contrast, for our SSCR, CTC can help train the generator better, which leads to defined objects. Furthermore, counterfactual reasoning also makes the predicted images more aligned to the instructions.

5 Conclusion and Future Work

We present a self-supervised counterfactual reasoning (SSCR) framework that introduces counterfactual thinking to cope with the data scarcity limitation for iterative language-based image editing. SSCR allows the model to consider new instruction-image pairs. Despite without ground-truth resulting images, we propose cross-task consistency (CTC) to provide a more explicit training signal and train these counterfactual instructions in a self-supervised scenario. Experimental results show that our counterfactual framework not only trains the image editor better but also improves the generalizability, even under data scarcity.

For the real world, both visual and linguistic will be more complicated. To accomplish real-world image editing, large-scale pre-trained language encoders and image generators should be applied to understand the diverse instructions and model the interaction for editing. From a theoretical perspective, our SSCR is a model-agnostic framework that can incorporate with any image generator, for GAN or non-GAN architecture, to perform real-world image editing. Currently, the interactive explainer and counterfactual intervention in SSCR both improve the editing quality in the token-level. To make it more suitable for real-world images, semantic-level intervention for the diverse natural instructions can support better counterfactual reasoning. Also, a stronger explainer that explains not only token-level error but also global editing operation between two images can provide robust self-supervised loss.

Acknowledgments. Research was sponsored by the U.S. Army Research Office and was accomplished under Contract Number W911NF-19-D-0001 for the Institute for Collaborative Biotechnologies. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Steven Bird and Edward Loper. 2004. NLTK: The Natural Language Toolkit. In *ACL*.

Jianbo Chen, Yelong Shen, Jianfeng Gao, Jingjing Liu, and Xiaodong Liu. 2018. Language-Based Image Editing with Recurrent Attentive Models. In *CVPR*.

Ming-Ming Cheng, Shuai Zheng, Wen-Yan Lin, Jonathan Warrell, Vibhav Vineet, Paul Sturgess, Nigel Crook, Niloy Mitra, and Philip Torr. 2013. ImageSpirit: Verbal Guided Image Parsing. In *ACM Transactions on Graphics*.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. In *NeurIPS WS*.

Alaaeldin El-Nouby, Shikhar Sharma, Hannes Schulz, Devon Hjelm, Layla El Asri, Samira Ebrahimi Kahou, Yoshua Bengio, and Graham W.Taylor. 2019. Tell, Draw, and Repeat: Generating and Modifying Images Based on Continual Linguistic Instruction. In *ICCV*.

Tsu-Jui Fu, Xin Wang, Matthew Peterson, Scott Grafton, Miguel Eckstein, and William Yang Wang. 2020. Counterfactual Vision-and-Language Navigation via Adversarial Path Sampling. In *ACL WS*.

Sahaj Garg, Vincent Perot, Nicole Limtiaco, Ankur Taly, Ed H. Chi, and Alex Beutel. 2019. Counterfactual Fairness in Text Classification through Robustness. In *AIES*.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2015. Generative Adversarial Networks. In *NeurIPS*.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *NeurIPS*.

Jin-Hwa Kim, Nikita Kitaev, Xinlei Chen, Marcus Rohrbach, Byoung-Tak Zhang, Yuandong Tian, Dhruv Batra, and Devi Parikh. 2019. CoDraw: Collaborative Drawing as a Testbed for Grounded Goal-driven Communication. In *ACL*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

Matt J. Kusner, Joshua Loftus, Chris Russell, and Ricardo Silva. 2017. Counterfactual Fairness. In *NeurIPS*.

Gierad Laput, Mira Dontcheva, Gregg Wilensky, Walter Chang, Aseem Agarwala, Jason Linder, and Eytan Adar. 2013. PixelTone: A Multimodal Interface for Image Editing. In *CHI*.

Takeru Miyato and Masanori Koyama. 2018. cGANs with Projection Discriminator. In *ICLR*.

Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, and Jason Yosinski. 2017. Plug & Play Generative Networks: Conditional Iterative Generation of Images in Latent Space. In *CVPR*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *ACL*.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. 2016. Generative Adversarial Text to Image Synthesis. In *ICML*.

Scott Reed, Aäron van den Oord, Nal Kalchbrenner, Sergio Gómez Colmenarejo, Ziyu Wang, Dan Belov, and Nando de Freitas. 2017. Parallel Multiscale Autoregressive Density Estimation. In *ICML*.

Neal J. Roese. 1997. Counterfactual Thinking. In *Psychological Bulletin*.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved Techniques for Training GANs. In *NeurIPS*.

Seitaro Shinagawa, Koichiro Yoshino, Sakriani Sakti, Yu Suzuki, and Satoshi Nakamura. 2017. Interactive Image Manipulation with Natural Language Instruction Commands. In *NeurIPS WS*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *CVPR*.

Fuwen Tan, Song Feng, and Vicente Ordonez. 2019. Text2Scene: Generating Compositional Scenes from Textual Descriptions. In *CVPR*.

Ronald J. Williams and David Zipser. 1989. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. In *Neural Computation*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. In *ICML*.

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong Hes. 2018. AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks. In *CVPR*.

Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. 2019. Self-Attention Generative Adversarial Networks. In *PMLR*.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. 2017. StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networkss. In *ICCV*.

Ran Zmigrod, Sabrina J. Mielke, Hanna Wallach, and Ryan Cotterell. 2019. Counterfactual Data Augmentation for Mitigating Gender Stereotypes in Languages with Rich Morphology. In *ACL*.