





[Re] Counterfactual Generative Networks

Ankit Ankit^{1, }, Sameer Ambekar^{1, }, Baradwaj Varadharajan^{1, }, and Mark Alence^{1, }

¹University of Amsterdam, Amsterdam, The Netherlands – ¹Equal contribution

Edited by
Koustuv Sinha

Reproducibility Summary

Reviewed by
Anonymous Reviewers

Scope of Reproducibility

In this paper, we attempt to verify the claims that the paper [sauer2021counterfactual] makes about their proposed CGN framework that decomposes the image generation process into independent causal mechanisms. Further, the author claims that these counterfactual images improves the out-of-distribution robustness of the classifier. We use the code provided by the authors to replicate several experiments in the original paper and draw conclusions based on these results.

Methodology

We use the same hyperparameters and architecture as mentioned in CGN [sauer2021counterfactual]. We use the PyTorch code from [the authors' publicly available repository](#). We make several changes to their code for the MNIST datasets since it gives spurious results/errors. Since we use ImageNet 1000 as a replacement for the ImageNet dataset, we modify the code accordingly. We reproduce tables 1-6 from CGN [sauer2021counterfactual] paper, excluding results for models from other papers.

Results

We validated each of the author's claim through the experiments given in the original paper and few additional experiments of our own. Overall, we found many experiments yielding identical results while some deviations were observed with both the Counterfactual Generative Network and the subsequent classification task. We were able to explain most of these deviations through our additional experiments while some couldn't be validated due to computational limitations.

What was easy

Overall, clear environment setup instructions, well working code and availability of pre-trained CGN models for both datasets proved valuable to validate the authors' claim.

Copyright © 2022 A. Ankit et al., released under a Creative Commons Attribution 4.0 International license.

Correspondence should be addressed to Ankit Ankit (ankitn1721@gmail.com)

The authors have declared that no competing interests exist.

Code is available at https://github.com/ambekarsameer96/FACT_AI/ – DOI 00.0000/zenodo.0000000.

swlh1:dir:aaaaaaaaaaaaaaaaaaaaaaaaaaaaa.

Open peer review is available at <https://openreview.net/forum?id=BSHg22G7n0F>.

What was difficult

Some experimental details were not reported in the original paper which made validations time consuming. ImageNet based experiments were replaced with ImageNet-1k(mini) due to the computational limitation which made it difficult to validate the author's original claims. Pre-trained classification models could have proven helpful in this case, but were unavailable, which meant we had to train the classifier from scratch. Code changes were required to obtain baseline results which was tedious considering different code architecture was implemented for MNIST & ImageNet.

Communication with original authors

We emailed the authors regarding inception score, MNIST dataset hyperparameters and ImageNet hyperparameters. We are awaiting a response from their end.

Code available at https://anonymous.4open.science/r/re_counterfactual_generative-E18F

1 Introduction

Neural Networks (NNs) have become ubiquitous in machine learning due to their predictive power. However, a shortcoming of NNs is their tendency to learn simple correlations that lead to good performance on test data rather than more complex correlations that generalise better. This shortcoming is apparent in the task of image classification, where NNs tend to overfit to factors like background or texture. To address this shortcoming, [sauer2021counterfactual] proposes a method of generating counterfactual images that prevent classifiers from learning spurious relationships.

The authors take a causal approach to image generation by splitting the generation task into independent causal mechanisms. The authors considered three separately learned Independent Mechanisms (IMs) to generate shapes, textures and backgrounds for an image. For the MNIST setting, all IM specific losses are optimized end-to-end from scratch, while in the ImageNet setting, each IM is initialized with weights from pre-trained BigGAN-deep-256[brock2019large]. The counterfactual image is then generated by passing the result of each IM to a deterministic composer function.

In this report, we use the publicly available code provided by the authors to reproduce the results of the paper and validate the authors' claims. In this endeavour, we made modifications to the code to determine the efficacy of their generative model and validate its impact on improving the out of distribution robustness of a classifier.

2 Scope of reproducibility

In this report, we investigate the following claims from the original paper:

1. Generating high-quality counterfactual images that decompose into independent causal inductive biases, these mechanisms disentangle object shape, object texture and background
2. Using counterfactual images improves the shape vs texture bias which is an inherent problem of deep classifiers
3. Using counterfactual images improve the out-of-distribution robustness for the classifier during the classification task
4. The Generative model can be trained efficiently on a single GPU with the help of powerful pre-trained models

We attempt to reproduce the experiments from the paper [sauer2021counterfactual] and perform exploratory analysis on the above mentioned claims. We propose using an extra loss function to mitigate some of the shortcomings during counterfactual generation process and generate heatmap plots to study the classifier behaviour.

3 Methodology

Alex et al. [sauer2021counterfactual] propose a Counterfactual Generative Network (CGN) framework to generate high-quality counterfactual images, which can be used to train invariant classifiers. The architecture of a CGN is composed of three IMs that are trained to generate backgrounds, shapes, and textures. Each IM is provided with a label. The task of the invariant classifier is to predict the label of a specific IM, regardless of the labels of the others. In conjunction with the composer function, the use of counterfactual images generated by the three IMs prevents the classifier from learning spurious relationships that arise from training on a natural dataset only.

The architecture of the CGN consists of a GAN as the backbone of each IM. Each IM samples random noise $\mu \sim N(0,1)$, along with an independently sampled label to generate

samples. The output x_{gen} is generated using an analytical function from the Composer 'C',

$$x_{gen} = C(m, f, b) = m \otimes f + (1 - m) \otimes b$$

where 'm' is the mask (alpha map), f is foreground and b is background. \otimes denotes the element wise multiplication.

The losses $\mathcal{L}_{rec}(x_{gt}, x_{gen})$, \mathcal{L}_1 reconstruction loss, $\mathcal{L}_{perceptual}$ as shown in Fig. ?? are used to improve the quality of generated images. Once the CGN is trained, u and y are randomized per mechanism such that new counterfactual x_{gen} are generated. Furthermore, hyperparameters such as CF ratio (the ratio indicates how many counterfactuals are generated per sampled noise) can be used to control the number of samples that are being generated. These samples are then used to train the classifier and evaluated on the corresponding test set.

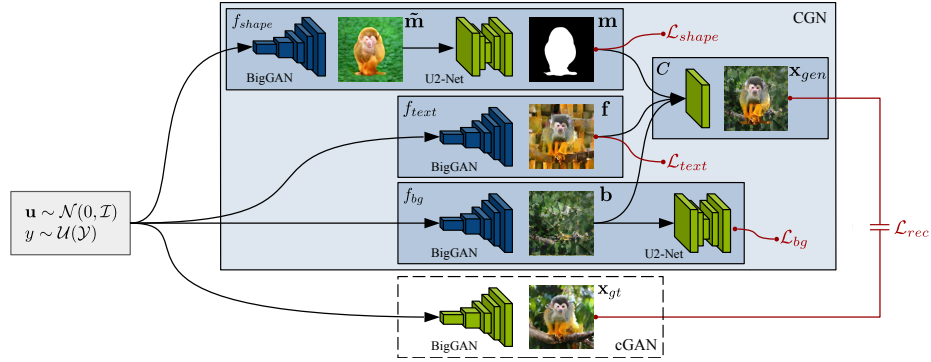


Figure 1. Architecture diagram from [sauer2021counterfactual] for ImageNet [imagenet] dataset. We observe that the architecture consists of f_{bg} , $f_{texture}$, f_{shape} to assist with the generation of x_{gen} . A powerful pre-trained Biggan-256 [brock2019large] is used to images from noise for each of the independent mechanisms. The shape and background are extracted with the help of a pre-trained U2-net [qin2020u2], while texture is obtained by minimizing perceptual loss between the foreground (f_{text} and a patch grid obtained from the value within the mask). The composer is analytically defined which uses alpha blending to generate the counterfactual x_{gen} . Components with trainable parameters are 'green' and without are 'blue'.

3.1 Model descriptions

The ImageNet variant follows the architecture that is illustrated in Fig. ?. The MNIST variant applies a simpler architecture by applying a second texture mechanism rather than a background mechanism.

3.2 Experimental setup and code

We use the datasets mentioned in [sauer2021counterfactual], excluding ImageNet [deng2009imagenet] due to limited resources and computational constraints.

For all the experiments, we make use of standard dataset splits akin to the CGN paper [sauer2021counterfactual]. Considering the computational constraint to train a classifier on ImageNet[imagenet], we used the pre-trained CGN to generate counterfactual images and trained a classifier on ImageNet-1k(mini) and mini-imagenet datasets.

3.3 Hyperparameter search

We found that the hyperparameters provided by the authors were stable, and so we did not conduct a hyperparameter search in this report.

⁰¹<https://kaggle.com/ifgotin/imagenetmini-1000>

Dataset	Description
Colored MNIST	Consists of digits in red or green.
Double Colored MNIST	Consists of more varied backgrounds and digits than Colored MNIST.
Wildlife MNIST	An attempt to build MNIST [lecun1998gradient] closer to the ImageNet[deng2009imagenet], texture was added as a bias to the data. The ten digits of the striped texture class encode the foreground labels and the background is labelled with the with the texture class 'veiny'.
ImageNet-1k(mini)	Subset of the ImageNet-1k[ImageNet-1k], available here ¹ that contains 34745 images in train set and 3923 for validation set, each split among 1000 classes individually.

Table 1. Datasets used

3.4 Computational requirements

All models are run on Nvidia GTX1080Ti GPUs (11Gb VRAM). For the MNIST datasets, training a CGN and a classifier each took approximately one hour.

4 Results

A lack of compute power prevented us from replicating the experiments on ImageNet. As a workaround, we limit ourselves to verifying the results using the ImageNet-1k(mini) dataset. This is beneficial because it extends the results of the paper and evaluates the method on a new dataset, and ensures that results can be reproduced with limited resources by referring to our report/code and the CGN paper.

4.1 Results reproducing original paper

Can Image generation process be decomposed into independent causal inductive biases effectively? – We begin the experiment by training a CGN on the three variants of the MNIST dataset. We observe in Fig. ?? that the digits in case of colored MNIST dataset lose their shape when reconstructed, whereas for double colored and wildlife MNIST, the digits look much better. Since we do not clearly understand why the shape in Colored MNIST is poor, we generated a mask timeline to verify any patterns. Fig. ?? details the same. Further analysis on this was conducted and recorded in ??. We also propose an additional loss function to help mitigate this problem.

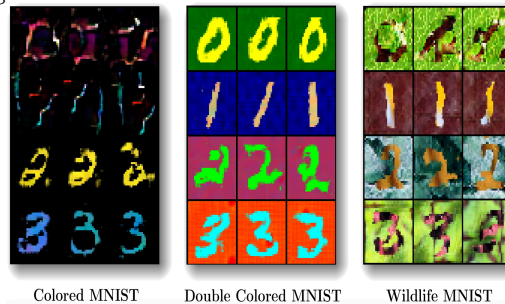


Figure 2. For brevity, we display only first 3 digits that were generated by training from scratch by us for the given three MNIST datasets.

Quality of Counterfactual Images on ImageNet-1k

To quantify the quality of the composite images produced by the CGN, the authors calculate the inception score (IS). The details of the IS calculations (inception model used, number of images used) were not mentioned in the paper. In an attempt to recreate the results regarding IS, we use the OpenAI implementation ¹. We plot the results of IS vs the number images using 10 splits in Fig. ?? . We observe the IS converges to an IS of 198.

We made use of the pre-trained CGN trained on ImageNet-1k that was present as part of the codebase to generate counterfactual images. Since there is no quantitative way to measure the quality of counterfactual images, we reproduced the images given in the original paper. We achieved a similar quality of counterfactual images but also noted deviations. Fig. ?? shows all the images that were given in the original paper. A deviation in the mask is observed for the class 'Agaric' and 'Cauliflower'. The difference in the images to the original paper prompted us to collect the classes with poorer counterfactual images to observe any patterns.

Fig. ?? is generated from the pre-trained CGN that have a low quality of images picked from random classes. Since the analysis is qualitative, we relied on the realism of the counterfactual compared to original images from that class. Images under the classes 'Cliff dwelling' 'American Chameleon' suffer from Texture-background entanglement resulting in the counterfactual with no subject. On the other hand, the images under the class 'Goldfinch', 'Junco' suffer from reduced realism due to linear constraints applied on the composer.

Impact of counterfactual images towards shape-bias of the classifier – Experiments conducted with ImageNet-1k(mini) dataset

In order to identify the impact of shape bias on the classifier, we made use of the proposed architecture for the classifier ensemble that included 3 different heads. The ensemble includes a pre-trained classifier (we made use of Resnet-50) as the backbone, while attaching 3 different heads to it. Each head controls the variance with respect to one of the 3 independent mechanism (Shape, Texture, Background) which are individually trained from scratch. The result from these heads are averaged to get the prediction accuracy of the classifier ensemble.

The results in Table ??(a) for ImageNet-1k(Mini) showed a considerable deviation. The shape bias is marginally lower compared to the baseline result while the texture bias is high. The reduction in the shape bias could be due to the smaller dataset that we are using. Since this is ambiguous to validate the original claim we conducted additional experiments which are detailed in section ??.

Do Counterfactual images improve the OOD robustness of the classifier? – Classification Accuracy (MNIST Dataset) Firstly, we trained a classifier on counterfactuals generated by the pre-trained CGN provided by the authors. It was not clear how many counterfactual images the classifier should be trained on, but the accuracies in Table ?? were similar to the results in the ablation study in Fig. 7 using 10^6 counterfactuals, so this is the number we chose. There was also ambiguity between the statements in the paper and the code about the classifier being trained on any real images, so we trained two classifiers. One classifier was shown real images, and the other was not.

The classifier trained with counterfactuals generated by the pre-trained models achieved comparable results to those in the paper. From table ??, it can be seen that the pre-trained models achieved train accuracies that differed by less than 3%, and test less than 1.5% compared to the results in the paper. However, the classifier trained on counterfactuals generated by CGNs that we trained (using the provided configurations) performed significantly worse on colored MNIST and wildlife MNIST in terms of test accuracy. We anticipate that the provided configurations were not the same as the configurations used to acquire the results in the paper.

¹<https://github.com/nnyyi/Inception-Score>

Dataset	Shape Bias	Dataset	IN-9	Mixed Same	Mixed Rand	BG-Gap
ImageNet-1k	48.1%	ImageNet-1k	17.27%	6.37%	7.65%	1.28%
ImageNet-1k + CGN/Shape	47.00%	ImageNet-1k	18.2%	14.05%	12.35%	1.7%
ImageNet-1k + CGN/Text	37.01%	+ CGN				
ImageNet-1k + CGN/Bg	47.02%					
(a) Impact on shape bias		(b) Out-of-distribution accuracy for ImageNet variants				
Dataset	Top-1 Train Accuracy	Top-5 Train Accuracy	Test Accuracy			
ImageNet-1k(mini)	91.27%		97.35%		73.12%	
ImageNet-1k(mini) + CGN	90.32%		97.24%		11.36%	
(c) Train and Test accuracies for ImageNet-1k(mini) with Resnet-50 backbone						

Table 2. Results for experiments conducted using Imagenet-1k(mini) dataset

	Colored MNIST		Double-colored MNIST		Wildlife MNIST
	Train Acc	Test Acc	Train Acc	Test Acc	Train Acc
Pre-Trained (Ours/With real images)	100.0	96.98	98.9	92.29	99.7
Pre-Trained (Ours/Without real images)	100.0	92.70	98.9	90.42	99.8
Trained (Ours/With real images)	98.7	68.96	96.8	88.54	99.9
Trained (Ours/Without real images)	98.7	43.88	96.7	87.90	99.9
Original+CGN (Theirs)	99.7	95.10	97.4	89.00	99.2

Table 3. MNIST Classification Accuracy

The presence of real images in the dataset for the pretrained models appeared not to have a significant effect on train or test accuracy. The largest gain obtained by including real images was approximately 4%. This demonstrates that the ambiguity regarding whether or not real images were used in the training of the classifier was inconsequential. For the CGNs that we trained, however, the presence of real images improved the performance of the classifier significantly.

Classification Accuracy(ImageNet Dataset)

The classifier was trained on counterfactual images from pre-trained CGN and ImageNet-1k(mini). The results in table ??(c) indicate the trend that was observed. The training accuracy showed a similar trend to the original paper’s classifier (trained on ImageNet). There is a similar drop in the training accuracy compared to the baseline(ImageNet-1k). Even though the original paper does not include the test accuracy for the classifier for the same distribution, we found that the classifier does not perform well with respect to the test data. The drop in top-1(the predicted class is the correct class that the image corresponds to) & top-5(5 out of 1000 classes with the highest probability as predicted by the classifier matches the actual label) accuracy compared to the baseline was attributed to the ability of the counterfactual models to reduce the shape bias of classifier which would improve the classifier’s robustness to unseen data. However, this is invalidated by the low percentage of the test accuracy. To further understand why the classifier ensemble is not performing well with unseen test data, we conducted additional experiments to explain the same behaviour.

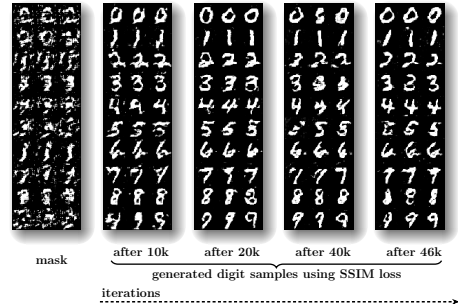
Out of distribution accuracy: A similar study as given in the paper was conducted to understand how the trained model performs with an out-of-distribution dataset. Table

??(b) contains the information with respect to the ImageNet-1k(mini) + CGN. There is a significant reduction in the accuracy of the out-of-distribution dataset. The baseline also showed a similar trend, and we could not achieve the higher percentage reported as part of the paper. We concluded that the baseline result is on the lower side primarily because of the size of the ImageNet-1k(mini) dataset that was used for training. Since the results show that the ensemble classifier improves the out-of-distribution robustness compared to the baseline, the percentage was still very low to make any conclusion. Both the trend with the test accuracy and out-of-distribution accuracy falls on the lower side, which prompted us to investigate further. We generated explainability plots using the same distribution and out-of-distribution data to determine how the model is behaving with and without the heads that disentangle shape, texture, background. We recorded All of the experiments as part of section ??.

4.2 Results beyond original paper

For Additional Result 1 we make use of CGN[sauer2021counterfactual] architecture that has been designed for MNIST datasets due to computational limitations.

Additional Result 1 - Does f_{bg} , $f_{texture}$, f_{shape} and $\mathcal{L}_{perceptual}$ (Perceptual loss) proposed in [sauer2021counterfactual] cover all aspects of background, shape, texture? – CGN [sauer2021counterfactual] makes use of texture loss $\mathcal{L}_{text}(x_{gt}, x_{gen})$, = sampling 36 patches of size 15 x 15 grid from regions wherever mask has values near 1. Further, from these 36 patches, a patch grid of 6 x 6 is used. It is then upsampled to 256 x 256 resolution, which is in turn used an input to the Perceptual loss $\mathcal{L}_{perceptual}$ between foreground f and patch grid $\mathcal{L}_{text}(f, pg)$. However, we observe that important image properties such as luminance, contrast, structure are not taken into consideration with the \mathcal{L}_{text} loss proposed in CGN [sauer2021counterfactual] for the generated image and the ground truth image and also because Hence, we propose the usage of an additional Loss function \mathcal{L}_{ssim} (SSIM) [wang2004image]. In addition, motivated by results as shown in [zhao2016loss], [pandey2020unsupervised] L2 loss unlike SSIM [wang2004image] over different distortions of the image remains constant instead of recognising them. It complements the structural loss \mathcal{L}_{rec} . Default Gaussian Kernel of 11 was used as a hyperparameter for SSIM [wang2004image]. We observe from Table ?? that using SSIM [wang2004image] loss improves classification accuracy on the Wildlife MNIST dataset. Qualitative improvements in the generated images can be seen in Fig ?. Images trained with SSIM [wang2004image] loss show better structure and crisper outlines. Improvements can be seen using SSIM [wang2004image] loss on the Double Colored MNIST dataset to a lesser extent. However, accuracy on the colored MNIST dataset decreases. This may be due to the dataset’s shape/structure/bias.



(a) Wildlife MNIST mask samples obtained using default hyper-parameters mentioned in CGN [sauer2021counterfactual]. (b) Wildlife MNIST mask samples obtained by adding SSIM [wang2004image] loss.

Figure 3. Results for experiments conducted using Wildlife MNIST dataset

Comparing Fig. ?? and Fig. ??, we observe that usage SSIM [wang2004image] leads to generation of mask samples that are sharper, capture more structure details, clearer outputs. Specifically, when compared to Fig. ?? the digits 0, 2, 4 lead to better visual outputs. As a result, we show in Table ?? that the overall classifier’s accuracy increases by around 16% when compared to training from scratch by us, and around 6% when compared to accuracy of given pre-trained model.