Learning Recourse on Instance Environment to Enhance Prediction Accuracy

Lokesh N

PhD student with IIT Bombay

Advised by Prof. Sunita Sarawagi and Prof. Abir De

Sep 23, 2022

Agenda

- Background on Causality
- Causal Motivation
- Image (Data) Recourse

Introduction

- Causal Inference is all about what is the effect of X on Y
- Machine learning is all about learning a function that navigates $X \to Y$
  - The goal is to apply f on unforeseen X

- Deep Learning is all about approximating the function $f_\theta : X \to Y$ using neural nets
- f_θ is only as good as the data we throw at it
- This begs the question: *What is the right data?*
  - Whatever makes $f_\theta(x_{test}) = y_{test}$

A Running Example for Causal Inference

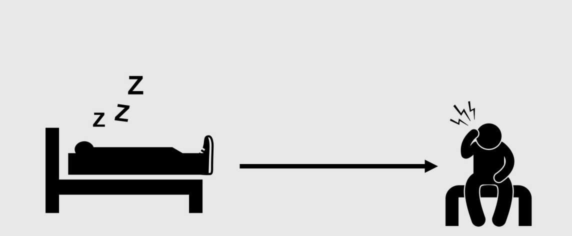


Figure: Impact of sleeping with shoes “ON” on headache the following morning. Credits - Brady Neal

Fundamental Problem in Causal Inference

Name	Covariates	Headache$\vert$Shoes	Headache $\vert\neg$shoes
A	α	Y	NaN
B	β	Y	NaN
C	γ	NaN	N
D	δ	NaN	N

ML aids Causal Inference

If we know to hallucinate the values of NaN in the observational data, computation of Causal effect is straight forward.

Machine learning is good at it!

These hallucinations are called counterfactuals.

- Before a binary event occurs, there are two potential outcomes $Y(0), Y(1)$. If the event 1 occurs, then $Y(0)$ is a counterfactual.
- There is no way to assess the correctness of estimated $Y(0)$ other than placing our bets on statistical significance of our belief on it.

Structural Causal Model - Key to Causal Inference

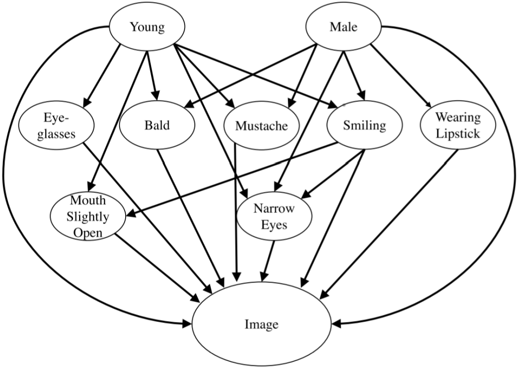


Figure: Causal GAN. Credits - M Kocaoglu et. al. 2017

SCM is not learnable from observational data

- We cannot learn an SCM just given the observational data
- In practice, it is specified by domain knowledge
- To be able to learn from observational data, we need to have the capacity to perform Randomized Control Trials

Generating Counterfactual Images given the SCM

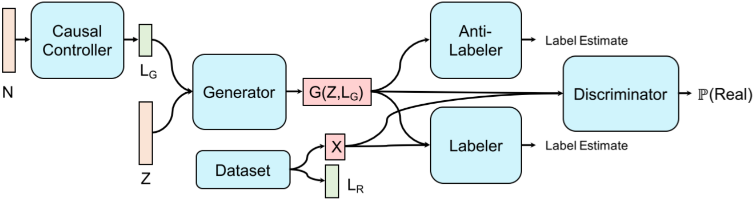


Figure: Causal GAN architecture. Image Credits: Causal GAN paper

Causal GAN objective:

$$
\begin{aligned}
min_G E_{x\sim p_{data}}\left[log(D(x))\right] + E_{x\sim p_g}\left[log\left(\frac{1-D(x)}{D(x)}\right)\right] \\
-\frac{1}{d}\sum_{j=1}^{d}\rho_j E_{x\sim P_g(x|l_j=1)}\left[log(D_{LR}(x)[j])\right] - (1-\rho_j)E_{x\sim P_g(x|l_j=0)}\left[log(1-D_{LR}(x)[j])\right] \\
+\frac{1}{d}\sum_{j=1}^{d}\rho_j E_{x\sim P_g(x|l_j=1)}\left[log(D_{LG}(x)[j])\right] - (1-\rho_j)E_{x\sim P_g(x|l_j=0)}\left[log(1-D_{LG}(x)[j])\right]
\end{aligned}
$$

Take it from this slide that generating counterfactual is not a

Causality inspired Algorithmic Recourse

We are interested in only Actionable recourse. *i.e.* recourse should not ask an individual to become 1 year younger to get desired label.

- Recourse > Explanations
- Beyond explanation, we need to show the path to obtain a better label
- Thus any recourse solves the following optimization problem

$$x^{CF} = argmin_{x'} cost(x, x' = h(x))$$

$$s.t. \;\; y(x) < y(h(x))$$

$$dist(x, h(x)) \leq \epsilon$$

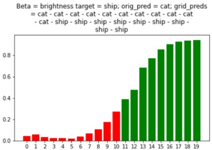
Hope

$h(x)$ suggests causal interventions because otherwise the one who deployed ML as a service can be exploited to the likes of her users. Eg. Adversarial perturbations.

Moving to the Project $\cdots$

Introduction & Motivation

- ML models tend to perform well on data that is *i.i.d.* sampled from train distribution and fail to generalize well to unseen instances.
- We want to train a recourse model alongside a classifier so that we know what is out of distribution and try to transform it to be in-distribution.



- Our motivation stems from the result we show left.
- For CIFAR dataset, we observe that for almost every misclassified image, as we vary a parameter (brightness), there is a *range* of values where the classifier emits the right label

An RL based baseline

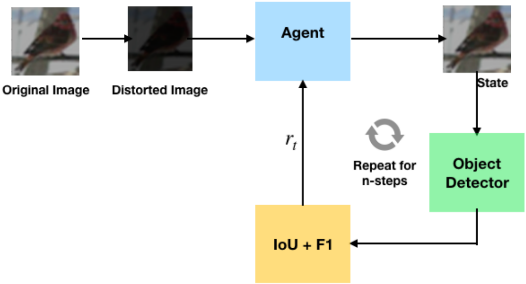
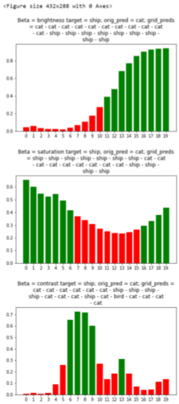


Figure: Improving Object detection through Reinforcement Learning.
Image credits: S Nayak et. al. 2020

Label space of the classifier is smooth

Probability of Ship

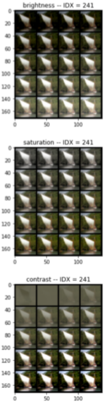


```
<Figure size 432x288 with 0 Axes>
```

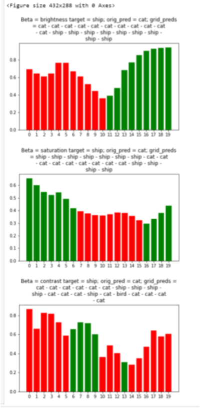
Beta = brightness target = ship; orig_pred = cat; grid_preds = cat - cat - cat - cat - cat - cat - cat - cat - cat - cat - cat - ship - ship - ship - ship - ship - ship - ship - ship - ship

Beta = saturation target = ship; orig_pred = cat; grid_preds = ship - ship - ship - ship - ship - ship - ship - cat - cat - cat - cat - cat - cat - cat - cat - cat - ship - ship - ship - ship

Beta = contrast target = ship; orig_pred = cat; grid_preds = cat - cat - cat - cat - cat - cat - ship - ship - ship - ship - cat - cat - cat - ship - cat - bird - cat - cat - cat - cat



Probability of \hat{y}



```
<Figure size 432x288 with 0 Axes>
```

Beta = brightness target = ship; orig_pred = cat; grid_preds = cat - cat - cat - cat - cat - cat - cat - cat - cat - cat - cat - ship - ship - ship - ship - ship - ship - ship - ship - ship

Beta = saturation target = ship; orig_pred = cat; grid_preds = ship - ship - ship - ship - ship - ship - ship - cat - cat - cat - cat - cat - cat - cat - cat - cat - ship - ship - ship - ship

Beta = contrast target = ship; orig_pred = cat; grid_preds = cat - cat - cat - cat - cat - cat - ship - ship - ship - ship - cat - cat - cat - ship - cat - bird - cat - cat - cat - cat

Insights so far

- For each image, there is a sweet spot of values of β where the likelihood of correct label is high
- If we know the range through a function (say g_ϕ), the classifier accuracy will be high!
- Thus we can think of such a function as providing recourse to the classifier
- What becomes more interesting is to learn both the recourse and classifier together

We can think of the above problem as a continuous armed bandit problem. Because the reward (probability of the correct label) is smooth, *perhaps* exploration is easy.

Proposed Solution

Learning Recourse on Instance Environment to Enhance Prediction Accuracy

The way we think about the problem

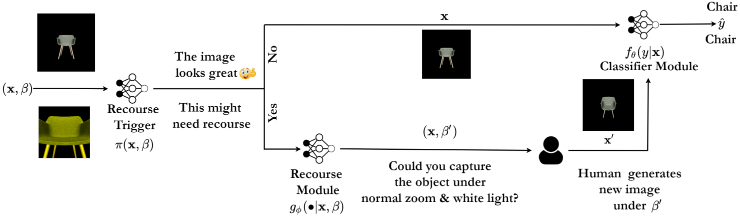


Figure: Architecture of Proposed Approach. The chair image on the top does not need recourse and attains the correct label from f_θ. However, the bottom image obtains the correct label only after recourse.

All ~~models~~ problems are good but some ~~are useful~~ have datasets

Dataset

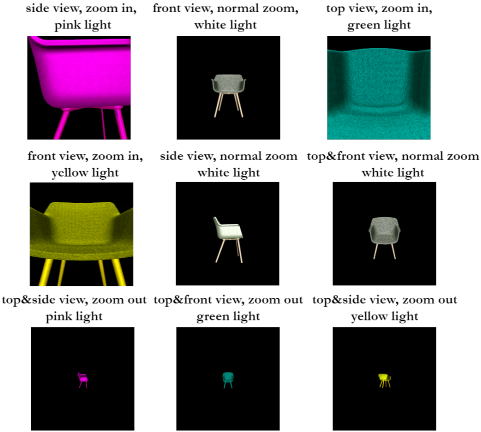


Figure: This figure shows renderings of a chair object under different βs. Each β is a 3-tuple namely *(view, zoom-level, light color)*.

- $z \in \mathcal{Z}$ represents a groud truth object
- $\boldsymbol{\beta} \in \mathcal{B}$ represents the instance enviroment
- $Z : \mathcal{Z} \times \mathcal{B} \rightarrow \mathcal{X}$ is the latent physical process (camera) that generates the images
- Medium of instruction to the user: $\mathcal{B}$
- Training Dataset $D = \{y_i, \{\mathbf{x}_{ij}, \boldsymbol{\beta}_{ij}\}_{j \in B_i}\}$

Solution Approach RecourseNet

We propose to tackle the problem by learning three modules. We perform a three stage sequential training approach. The three modules are.

- A classifier $f_\theta : \mathcal{X} \times \mathcal{Y} \to [0, 1]$
- A recourse trigger network $\pi : \mathcal{X} \times \mathcal{B} \to \{0, 1\}$
- A recourse recommender network $g_\phi : \mathcal{X} \times \mathcal{B} \times \mathcal{B} \to [0, 1]$

Learning Objective

$$\max_{\theta,\phi,\pi} \sum_{\substack{i\in D\\ j\in B}} \log \Bigg[\big(1-\pi(\mathbf{x}_{ij},\boldsymbol{\beta}_{ij})\big)\, f_\theta(y_i \mid \mathbf{x}_{ij}) + \pi(\mathbf{x}_{ij},\boldsymbol{\beta}_{ij}) f_\theta(y_i \mid Z(z_i, \operatorname{argmax}_{\boldsymbol{\beta}} g_\phi(\boldsymbol{\beta} \mid \mathbf{x}_{ij},\boldsymbol{\beta}_{ij}))) \Bigg] \tag{1}$$

$$\text{subject to, } \sum_{i\in D, j\in B} \pi(\mathbf{x}_{ij}) \leq b, \tag{2}$$

$$\pi(\mathbf{x}_{ij},\boldsymbol{\beta}_{ij}) \in \{0,1\} \tag{3}$$

Learning Recourse aware f_θ

- Training f_θ on entire training data may be suboptimal especially when some bad instances will be recoursed at test time.
- We first greedily select the instances that might be recoursed and then train the classifier on the remaining instances.

$$\mathrm{Recourse}_\Delta(\theta, \mathbf{x}_{ij}, y_i) = \{\boldsymbol{\beta}' \in B_i \mid \log f_\theta \left(y_i \mid Z(z_i, \boldsymbol{\beta}')\right) > \log f_\theta \left(y_i \mid \mathbf{x}_{ij}\right) + \Delta\} \tag{4}$$

and then we pose the following training problem to learn θ.

$$\max_{\theta,\pi} \sum_{\substack{i \in D \\ j \in B_i}} \left[\left(1 - \pi(\mathbf{x}_{ij}, \boldsymbol{\beta}_{ij})\right) \log f_\theta(y_i \mid \mathbf{x}_{ij}) + \pi(\mathbf{x}_{ij}, \boldsymbol{\beta}_{ij}) \max_{\boldsymbol{\beta}_{ir} \in \mathrm{Recourse}_\Delta(\theta, \mathbf{x}_{ij}, y_i)} \log f_\theta(y_i \mid \mathbf{x}_{ir}) \right]$$

$$\text{subject to,} \sum_{i \in D, j \in B_i} \pi(\mathbf{x}_{ij}, \boldsymbol{\beta}_{ij}) \leq b, \text{ and } \pi(\mathbf{x}_{ij}, \boldsymbol{\beta}_{ij}) \in \{0, 1\}. \tag{5}$$

Next we learn g_ϕ for a trained f_θ

We take note that g_ϕ has a role to play only when $\pi = 1$. In otherwords, when an input instance $(\mathbf{x}, \boldsymbol{\beta})$ is bad.

But how do we know what is bad, and for those that are bad how do we search for good examples?

A simple idea: In a group of B_i examples for the object i, why not make the least loss example the good one and bottom δ examples the bad ones?

Noisy Supervision

We can immediately guess that such a heuristic will lead to a lot of noise in supervision provided to g_ϕ.

We have this as a baseline and we will see that this baseline does not work well.

Our Solution Approach for g_ϕ

Idea:

- If $g_\phi(\mathbf{x}, \boldsymbol{\beta}) = \boldsymbol{\beta}'$, then $f_\theta(\mathbf{x}')[y]$ better be high
- But what if $\mathbf{x}' \notin D$? We definitely do not want to generate counterfactual images.

But we need to hallucinate something to make progress

- We hallucinate the probability of correctness of $\mathbf{x}'$ from f_θ had it been $\in D$
- Our counterfactual confidence is given by:

$$f^{\mathrm{CF}}(y|\mathbf{x}, \boldsymbol{\beta}) = \frac{\sum\limits_{(i,j) \in V : y_i = y, \boldsymbol{\beta}_{ij} = \boldsymbol{\beta}} f_{\hat{\theta}}(y_i = y \mid \mathbf{x}_{ij})}{\sum\limits_{(i,j) \in V : y_i = y, \boldsymbol{\beta}_{ij} = \boldsymbol{\beta}} 1} \tag{6}$$

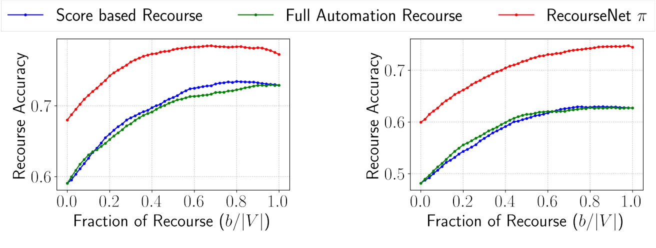
Let D_δ be the set of *groups* of examples $\in D$ that have atleast one example in the group with loss $< \delta$. $D_\delta = \{i | min_j \;\; loss_{ij} < \delta\}$
Then we learn g_ϕ using the following objective:

$$\begin{aligned}\max_{\phi} \sum_{i \in D_\delta} \sum_{j \in B_i} \max_{r \in B_i} \log \left[f_\theta(y_i \,|\, \mathbf{x}_{ir}) \, g_\phi(\boldsymbol{\beta}_{ir} \,|\, \mathbf{x}_{ij}, \boldsymbol{\beta}_{ij}) \right] \\ + \sum_{i \notin D_\delta} \sum_{j \in B_i} \log g_\phi \left(\mathrm{argmax}_{\boldsymbol{\beta}} \, f^{\mathsf{CF}}(y_i \,|\, \mathbf{x}_{ij}, \boldsymbol{\beta}) \,|\, \mathbf{x}_{ij}, \boldsymbol{\beta}_{ij} \right) \qquad (7)\end{aligned}$$

Computation of π

$$\pi(\mathbf{x}_{ij}, \boldsymbol{\beta}_{ij}) = [f^{\mathsf{CF}}(y_{\max} \mid \mathbf{x}_{ij}, \boldsymbol{\beta}'_{ij}) > f_{\widehat{\theta}}(y_{\max,i} \mid \mathbf{x}_{ij})] \tag{8}$$

Recourse Performance

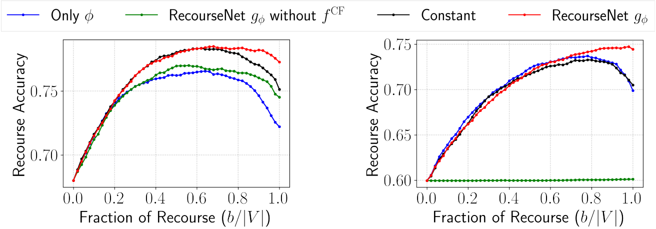


(a) Shapenet-Large

(b) Shapenet-Small

Figure: Variation of classification accuracy after recourse against the budget *b*, , the maximum number of instances selected for recourse for both Shapenet-Large and Shapenet-Small datasets for the recourse trigger π provided by RecourseNet, score based recourse and full automation recourse.

Recourse Recommender Performance

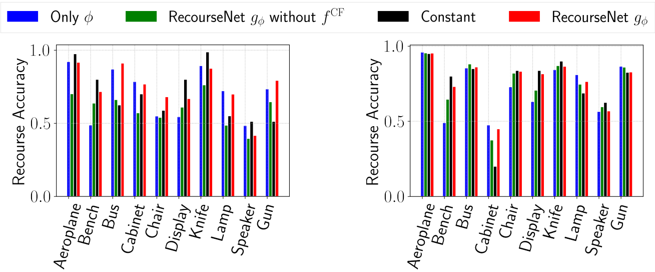


(a) Shapenet-Large.

(b) Shapenet-Small.

Figure: Variation of classification accuracy after recourse against the budget *b*, , the maximum number of instances selected for recourse for both Shapenet-Large and Shapenet-Small datasets for the recourse recommender g_ϕ provided by RecourseNet, Only ϕ, RecourseNet with f^{CF} and Constant.

Why is constant not appreciable?



(a) Shapenet-Large

(b) Shapenet-Small

Figure: Accuracy of different recourse recommenders for different classes.

Moving to our current problem (work in progress)

Dataset CRUNCH !

- The key to our earlier solution is that we have access to B_i examples for many i
- In practice, we may not have that.
- Perhaps, it is nice to make an alternative assumption: User will give us a small dataset D of her likes and we can additionally ask for b instances of our likes in an iterative manner
- Then, can we solve RecourseNet? Now, what does it mean for us to judiciously exhaust the query budget?
- Also can we solve RecourseNet effectively when $\boldsymbol{\beta}$ is continuous?

A glimpse on Recoursed images (Skin-Lesion detection task)

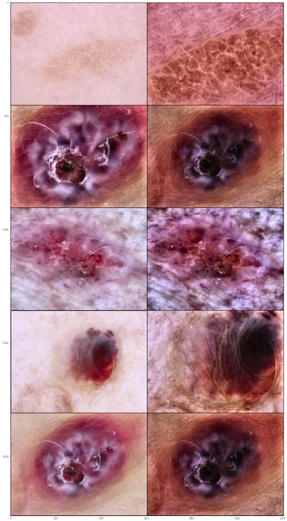


Figure: Skin Lesion Detection.

Thank You!