Meaningfully debugging model mistakes using conceptual counterfactual explanations.


CNS Seminar, Braian O. Dias
01/11/2022
Agenda

- Introduction
- Motivation
- Methods
- Results
- Conclusion
Introduction

**Debugging** ML models is an arduous task, the more complex the model the worse it gets.

- Usually an *ad hoc* process that explains model mistakes
- Outputs of this process might not be meaningful or not easy to understand
  - I.e.: CNN feature maps might help, but require human interpretation
- There’s no widely used systematic approach
Understanding and explaining mistakes made by a trained model is important to many ML objectives such as:

- Improve robustness
- Addressing concept drift
- Mitigating bias
The paper proposes a **systematic approach**.

**Conceptual counterfactual explanations (CCE)**

That “**explains why a classifier makes a mistake on a particular test sample(s) in terms of human-understandable concepts**.”
Introduction

The paper proposes a systematic approach,

**Conceptual counterfactual explanations (CCE)**

That “explains why a classifier makes a mistake on a particular test sample(s) in terms of human-understandable concepts”.

Human-understandable concept: a zebra is misclassified as a dog because of faint stripes.
Motivation

ML practitioners need to understand why a model is making a mistake.

Common questions:

- Was this kind of image **underrepresented** in my training distribution?
- Am I **preprocessing** the image correctly?
- Has my model learned a **spurious correlation or bias** that is hindering generalization?
Example 1: Usability of a Pretrained Model.

A pathologist uses an pretrained model to classify histopathology images.

The original documentation reports high accuracy, however on his data the model performs poorly.

After investigation, he finds that the hues of his images are different from the original data.

**Cause:** Domain shift (hues)
Example 2: Discovering bias

A doctor(1) collects images of skin diseases from his patients and trains a model to classify it.

The model performs well for doctor(1), however when shared to another doctor(2) from a different hospital, it starts to make many mistakes.

Doctor(1) realizes age and skin color of doctor(2)’s patients are different.

**Cause:** Bias in data (age, skin color)
CCE is a systematic method for explaining model mistakes using meaningful human-understandable concepts (hues, skin color...)

- No training data or retraining needed
- Only needs access to the model
- High-level explanations that are easy for users to understand
Motivation

Inspiration

**Counterfactual explanations:** class of explanation that provides a link between what could have happened had input to a model been changed in a particular way (Verma et al. (2020)).

**Concept Activation Vectors (CAV):** Linear classifiers trained in the bottleneck of a network and correspond to **concepts** (Kim et al., 2018). Explanations produced by CAV are high-level human-understandable concepts.
There are 3 key steps of CCE:

1. Define a **concept library**
2. Learn a CAV for each concept
3. Generate **perturbation in the embedding space** for a given misclassified test sample that would correct the model prediction.
Methods

There are 3 key steps of CCE:

1. Define a **concept library**
2. Learn a CAV for each **concept**
3. Generate **perturbation in the embedding space** for a given misclassified test sample that would correct the model prediction.
1. Define a concept library

Concept library: Set of human-interpretable concepts $C = \{c_1, c_2, c_3, \ldots\}$

I.e.: mirror, person, street, water, snow, stripes, ...

For each concept we collect:

- $P_{ci}$: Positive examples that contain the concept
- $N_{ci}$: Negative examples that does not contain the concept

Concepts can be user-defined or learned automatically from the data (Ghorbani et al., 2019)
1. Define a **concept library**

170 concepts adapted from **Broden**: Broadly and Densely Labeled Dataset (Fong & Vedaldi, 2018)

The data used to learn concepts can **differ** from the data used to train the model we apply CCE!

Figure 2. Samples from the **Broden** Dataset. The ground truth for each concept is a pixel-wise dense annotation.

2. Learn a CAV for each concept

We train a SVM to produce the corresponding CAV for each concept.

- Only done once for each model we want to evaluate.

In the experiments $f$ is a ResNet18 pretrained on ImageNet

```
Algorithm 2 Learning concept vectors
Inputs:
  f  trained network: model
  L bottleneck layer (hyperparameter): int
  concepts set of concepts: set[str]
  P # positive examples per concept: dict[str, list[sample]]
  N # negative examples per concept: dict[str, list[sample]]

Output:  svms #Set of SVMs containing concept predictors.

b, t = f.layers[:L], f.layers[L:]
# Divide network $f(\cdot)$ into a bottom $b$ (first $l$ layers) and top $t$ (remaining layers) so that $f(\cdot) = t(b(\cdot))$

for c in concepts:  # Per concept, learn an SVM to classify bottleneck representations of positive and negative examples.
  svms[c] = svm.train(b(P[c]), b(N[c]))
  # Filter out concepts that are not learned well
  if svms[c].acc < .7:
    del svms[c]
```
Methods

3. Conceptual Counterfactual Explanations

Optimization process that perturbs the input by varying the amount of different concepts, following three principles:

1. **Correctness**: A counterfactual is correct if it achieves the correct label
2. **Validity**: Don’t violate real-world conditions
3. **Sparsity**: Generalizing explanations. Perturbations should change a small number of concepts
Methods

$C^-$: concept vectors scaled by the maximum margin to the decision boundary

$\mathcal{L}_{CE}$: Minimize Cross Entropy loss, until correct prediction (correctness)

$w_{\min}, w_{\max}$: validity constraints

$\alpha |w|_1 + \beta |w|_2$: Elastic net regularization (sparsity)
Methods

A large **positive** score means that **adding/increasing** that concept will correctly classify the image.

A large **negative** score means that **removing/decreasing** that concept will correctly classify the image.

CCE offers us an assessment of which concepts explain a misclassified sample.
Introduction

Methods

a) Learning a Concept Bank

\[ b_L(z_1), b_L(z_2), b_L(z_3) \]
\[ c_{\text{stripes}}, c_{\text{cow}}, c_{\text{water}}, c_{\text{field}} \]
\[ C \in \mathbb{R}^{N_c \times m} \]

b) Generating Conceptual Counterfactuals

\[ b_L(z_1) + wC \]
\[ b_L(z_2) \]
\[ b_L(z_3) \]

Correct label → Zebra

African Hunting Dog

Stripes
Cow
Polka Dots
Dog

1.00
0.52
-0.2
-0.35

C

wC → Concept explanation
CCE to explain model limitations

**Spurious Correlations (High-level):** A classification model might learn correlations related to other objects present in the images other than the main one.

**Experiments:** 20 different training scenarios, each one containing 5 animal classes and one *confounding* variable - forcing a *spurious correlation*. A *control* model is also trained using random samples of animals in different contexts.

**Metric:** *Precision@3* - the spurious concept is among the top 3 concepts.
Spurious Correlations (High-level): A classification model might learn correlations related to other objects present in the images other than the main one. Fig 2.(a) from the paper.
CCE to explain model limitations - Spurious Correlations (High-level)

- Conceptual Sensitivity Score (CSS) - Baseline
- CoCoX (short for Conceptual and Counterfactual Explanations)
- CCE(Univariate) - simpler version

<table>
<thead>
<tr>
<th>Method</th>
<th>Mean Prec@3</th>
<th>Median Rank</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>0.02</td>
<td>82.65 (42.7, 120.4)</td>
</tr>
<tr>
<td>CSS</td>
<td>0.003</td>
<td>76.5 (69.79, 87.51)</td>
</tr>
<tr>
<td>CoCoX</td>
<td>0.73</td>
<td>4.63 (3.82, 5.89)</td>
</tr>
<tr>
<td>CCE(Control)</td>
<td>0.04</td>
<td>32.3 (28.03, 40.05)</td>
</tr>
<tr>
<td>CCE(Univariate)</td>
<td>0.91</td>
<td>2.00 (1.71, 2.35)</td>
</tr>
<tr>
<td>CCE</td>
<td>0.95</td>
<td>1.85 (1.80, 2.10)</td>
</tr>
</tbody>
</table>
CCE to explain model limitations - Low-level Image Artifacts

ImageNet has a green apple class called **Granny Smith**.

- Model explains that, as the image is **grayed**, the class prediction **decreases** while the CCE for **greenness** **increases**.
Results

CCE in the wild - Skin condition classification

Figure 4. CCE explains model mistakes using learned biases and image quality conditions. (a) CCE identifies dark skin type correlation with the allergic contact dermatitis condition that exists in the training dataset. (b, c, d) CCE identifies image artifacts that degrade the model performance.
Conclusion

- Offers **meaningful, human-understandable insights** for understanding and explaining mistakes in deep learning models.
- Results show CCE is **robust**, and can provide clues even when the concept does not exist in the concept database (e.g.: bed->sofa).
- It is **fast** (<0.3s on CPU).
- Doesn’t require access to the training data. **Concepts** can come from a separate dataset.
- Incorporating **automatic concept learning** might make CCE easier to use.