Measuring Representational Robustness of Neural Networks Through Shared Invariances

Vedant Nanda
University of Maryland & MPI-SWS

ICML 2022
Robustness in Deep Learning

“Desired” Invariances: $\mathcal{T} = \{ T_i | f(T_i(x)) = f(x) \}$

Prior works (Szegedy et al., 2013; Recht et al., 2020):

“Desired” = set of input transformations that do not change output for a human

Robustness is inherently relative*!

* w.r.t. “human”
What if $\mathcal{T}$ is defined by another Neural Network?

“Desired” Invariances: $\mathcal{T} = \{ T_i | f(T_i(x)) = f(x) \}$

“Desired” = input transformations that do not change output for a reference Network $m_1$

Robustness evaluation = measuring shared invariances between $m_1$ & $m_2$

A more granular view on robustness – depends on reference model $m_1$
Why define $\mathcal{T}$ using another Neural Network?

- Instead of having to rely on approximations of human perception (e.g., class labels), we have full access to representations of reference neural network.

- Allows us to investigate interesting questions about Deep Learning.

- Useful for a future society with multiple agents (e.g., driverless cars) controlled by neural nets.
Problem statement

Given a Neural Network $m_2 : \mathbb{R}^m \rightarrow \mathbb{R}^{d_2}$,

Reference Network $m_1 : \mathbb{R}^m \rightarrow \mathbb{R}^{d_1}$,

Inputs ($X \in \mathbb{R}^{n \times m}$),

How to measure shared invariances of $m_2$ w.r.t. $m_1$ given some $X$, i.e.,

$0 \leq S_{inv}(m_2 \mid m_1, X) \leq 1$

Such a measure $S_{inv}$ would be directional by design

$S_{inv}(m_2 \mid m_1, X) \neq S_{inv}(m_1 \mid m_2, X)$
Can’t we just use representation similarity measures?

A lot of prior work on measuring representation similarity ($S_{rep}$)

- SVCCA (Raghu et al., 2017), PWCCA (Morcos et al., 2018), CKA (Kornblith et al., 2019)

Generally, $S_{rep}$ takes two sets of representations and gives a similarity score, i.e.,

for $Y \in \mathbb{R}^{n \times d_1}$ and $Z \in \mathbb{R}^{n \times d_2}$, $0 \leq S_{rep}(Y, Z) \leq 1$

Problems:

- $S_{rep}$ are designed to measure correlations, invariances require interventions
- Interventions require exploring points outside $X$
- Generally not directional
Our proposal to measure shared invariance

1. Find *Identically Represented Inputs (IRIs)* $X, X'$ such that $m_1(X) \cong m_1(X')$

Representation Inversion \cite{Mahendran:2015:CVPR}

$$\argmin_{x'} \mathcal{L}(x')$$

$$\mathcal{L}(x') = \| m_1(x) - m_1(x') \|_2$$

$$x' = x' - \alpha \nabla_{x'} \mathcal{L}$$
Our proposal to measure shared invariance

1. Find *Identically Represented Inputs (IRIs)* $X, X'$ such that $m_1(X) ≡ m_1(X')$

2. Measure similarity of $m_2(X)$ and $m_2(X')$

**Representation Similarity Measures $S_{\text{rep}}$**

$$S_{\text{inv}}(m_2 \mid m_1, X, S_{\text{rep}}) = S_{\text{rep}}(m_2(X), m_2(X'))$$
Our proposal to measure shared invariance

1. Find *Identically Represented Inputs (IRIs)* $X, X'$ such that $m_1(X) \cong m_1(X')$

2. Measure similarity of $m_2(X)$ and $m_2(X')$

$S_{\text{inv}}(m_2 \mid m_1, X, S_{\text{rep}}) = \text{STIR}(m_2 \mid m_1, X) = \text{LinearCKA}(m_2(X), m_2(X'))$

(Kornblith et al., 2019)

- STIR explicitly captures invariances via *IRIs*
- STIR explores regions outside $X$
- STIR is directional
STIR offers insights beyond representation similarity

2 ResNet18 trained on CIFAR10 using same hyperparams – only differ in initial weights

|                  | $m_1| m_2$ | $m_2| m_1$ |
|------------------|----------|----------|
| **STIR**         | 0.605 ± 0.013 | 0.562 ± 0.023 |
| **CKA**          | 0.967 ± 0.000 |          |

Models trained using Vanilla loss only have **moderate** levels of shared invariance

|                  | $m_1| m_2$ | $m_2| m_1$ |
|------------------|----------|----------|
| **STIR**         | 0.934 ± 0.003 | 0.939 ± 0.002 |
| **CKA**          | 0.937 ± 0.000 |          |

Models trained with Adversarial Training (AT) have **high** levels of shared invariance

Models trained with AT should have higher shared invariance than Vanilla since AT induces invariance to $\ell_p$ perturbations

CKA is high in both cases and hence does not offer such insights
How do losses & network architectures impact STIR?

<table>
<thead>
<tr>
<th>m_1 (reference)</th>
<th>Vanilla, vgg19</th>
<th>Vanilla, vgg16</th>
<th>Vanilla, resnet34</th>
<th>Vanilla, resnet18</th>
<th>AT, vgg19</th>
<th>AT, vgg16</th>
<th>AT, resnet34</th>
<th>AT, resnet18</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vanilla, vgg19</td>
<td>0.32 0.26 0.21 0.14</td>
<td>0.38 0.26 0.30 0.28</td>
<td>0.44 0.36 0.28 0.28</td>
<td>0.38 0.41 0.29 0.29</td>
<td>0.78 0.78 0.84 1.00</td>
<td>0.85 0.82 1.00 0.82</td>
<td>0.94 1.00 0.93 0.92</td>
<td>1.00 0.94 0.93 0.92</td>
</tr>
<tr>
<td>Vanilla, vgg16</td>
<td>0.38 0.26 0.30 0.28</td>
<td>0.41 0.34 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>1.00 0.68 0.42 0.42</td>
<td>1.00 0.68 0.43 0.42</td>
<td>0.60 1.00 0.43 0.34</td>
<td>1.00 0.68 0.42 0.42</td>
<td>1.00 0.68 0.43 0.42</td>
</tr>
<tr>
<td>Vanilla, resnet34</td>
<td>0.44 0.36 0.28 0.28</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
</tr>
<tr>
<td>Vanilla, resnet18</td>
<td>0.38 0.41 0.29 0.29</td>
<td>0.41 0.39 0.35 0.39</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
<td>0.60 1.00 0.43 0.34</td>
</tr>
</tbody>
</table>

Penultimate Layer

**Loss:** Adversarial training leads to higher STIR scores – even for other architectures

**Architectures:** Higher STIR when residual networks are reference models for both Vanilla and AT!

Potential reason: initial layers have high shared invariances and residual connections preserve these features
How do training datasets impact STIR?

Also investigated by prior representation similarity works (Kornblith et al, 2019)

Final layers learn dataset specific features and hence have almost zero STIR

We find similar trend as CKA, however drop-off for STIR is higher in later layers
Additional questions investigated

Different initializations’ impact on STIR

AT consistently has higher STIR scores than Vanilla

Different adversarially robust losses’ impact on STIR

Only moderate STIR despite similar $\ell_p$ robustness!

How does more training data impact STIR

STIR monotonically increases as we add more data, but with diminishing returns

Check out the paper for details! tinyurl.com/stir-paper
Summary

Proposed **Similarity Through Inverted Representations (STIR)** to study relative invariances

Explicitly captures invariances via *Identically Represented Inputs*

Used STIR to investigate the impact of choices in a DL pipeline

AT consistently leads to higher STIR than Vanilla; residual connections increase STIR

---

Paper: tinyurl.com/stir-paper
Code: github.com/nvedant07/stir

Thank You!
vedant@cs.umd.edu
How Do Model Updates Change Shared Invariances? (backup)

Vanilla

Adversarial Training

STIR monotonically increases as we add more data, however, after a certain point, we see diminishing returns.