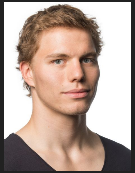
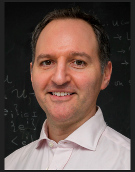
Measuring Representational Robustness of Neural Networks Through Shared Invariances

Vedant Nanda
University of Maryland & MPI-SWS

ICML 2022







Till Speicher

Camila Kolling

Krishna P. Gummadi

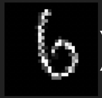
John P. Dickerson

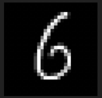
Adrian Weller

MAX PLANCK INSTITUTE FOR SOFTWARE SYSTEMS

Robustness in Deep Learning

"Desired" Invariances: $\mathcal{T} = \{ \boldsymbol{T}_i \mid f(\boldsymbol{T}_i(x)) = f(x) \}$

$f($  $) = f($  $)$ 

$f($  $) = f($  $)$ 

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: $\mathcal{T} = \{\ \boldsymbol{T}_i \mid f(\boldsymbol{T}_i(x)) = f(x)\ \}$

"Desired" = input transformations that do not change output for a *reference Network* $\boldsymbol{m}_1$

Robustness evaluation = measuring shared invariances between $\boldsymbol{m}_1$ & $\boldsymbol{m}_2$

A more granular view on robustness – depends on reference model $\boldsymbol{m}_1$

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

Architectures & Losses

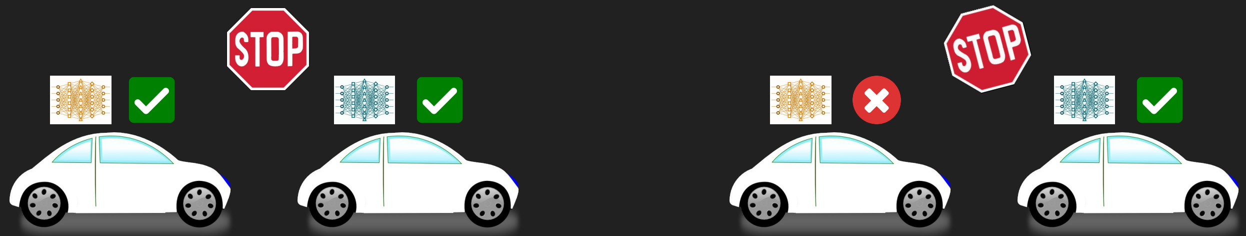
Training Dataset

Shared Invariance

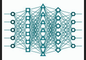
Weight Initialization

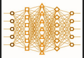
Model Updates

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



Problem statement

Given a Neural Network $\boldsymbol{m}_2$  : $\mathbb{R}^{m} \rightarrow \mathbb{R}^{d2}$,

Reference Network $\boldsymbol{m}_1$  : $\mathbb{R}^{m} \rightarrow \mathbb{R}^{d1}$,

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

How to measure shared invariances of $\boldsymbol{m}_2$ w.r.t. $\boldsymbol{m}_1$ given some $\boldsymbol{X}$, i.e.,

$0 \leq S_{inv}(\boldsymbol{m}_2 \mid \boldsymbol{m}_1, \boldsymbol{X}) \leq 1$

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

$$S_{inv}(\boldsymbol{m}_2 \mid \boldsymbol{m}_1, \boldsymbol{X}) \neq S_{inv}(\boldsymbol{m}_1 \mid \boldsymbol{m}_2, \boldsymbol{X})$$

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 $\boldsymbol{Y} \in \mathbb{R}^{n \times d1}$ and $\boldsymbol{Z} \in \mathbb{R}^{n \times d2}$, $0 \leq S_{rep}(\boldsymbol{Y}, \boldsymbol{Z}) \leq 1$

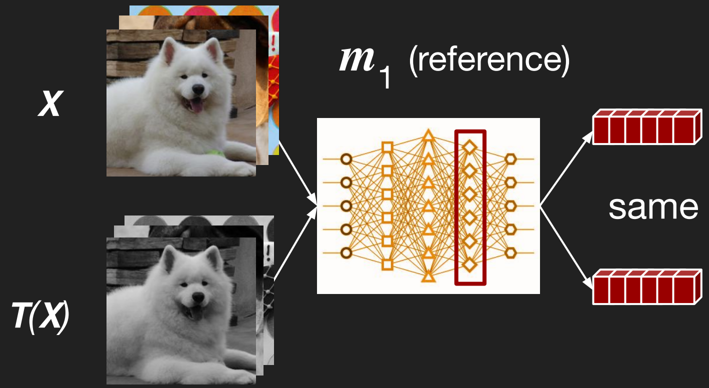
Problems:

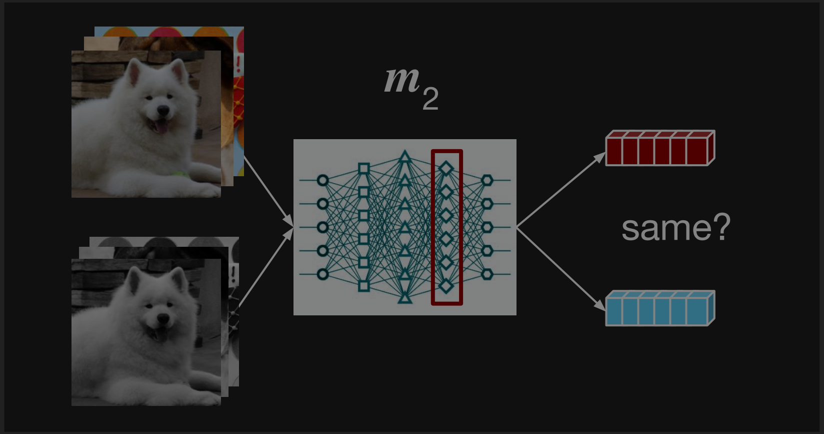
S_{rep} are designed to measure correlations, invariances require interventions

Interventions require exploring points outside X

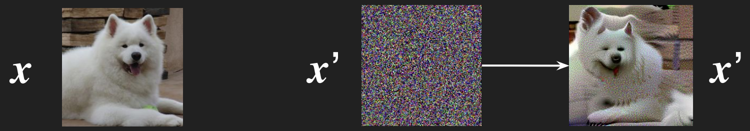
Generally not directional

Our proposal to measure shared invariance





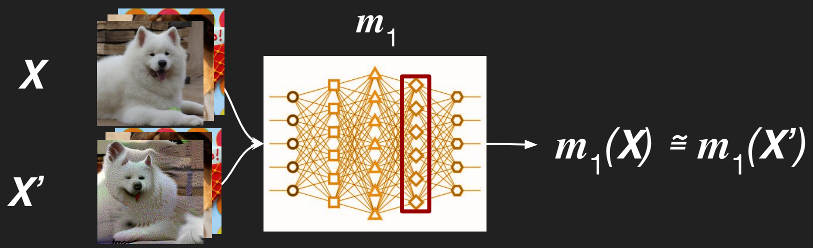
1. Find *Identically Represented Inputs (IRIs)*
   X, X' such that $m_1(X) \cong m_1(X')$



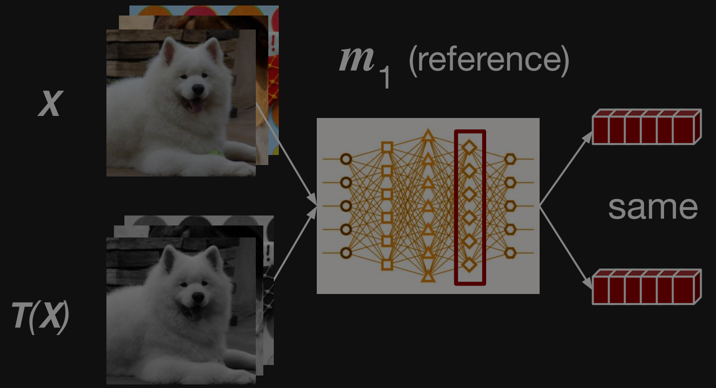
Representation Inversion (Mahendran & Vedaldi, CVPR 2015)

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

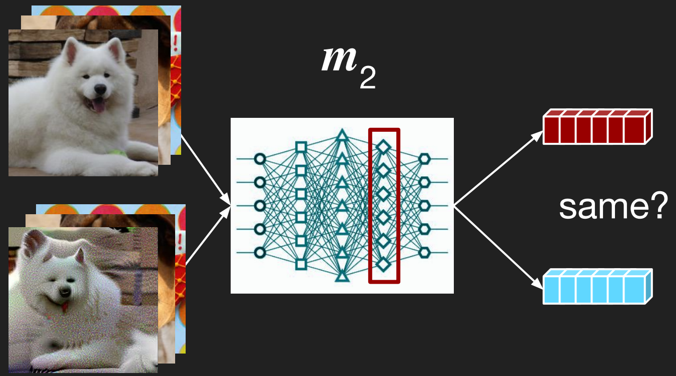
$$x' = x' - \alpha \nabla_{x'} \mathcal{L}$$



Our proposal to measure shared invariance

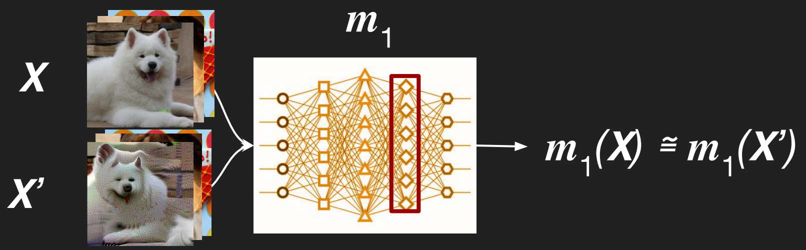


1. Find *Identically Represented Inputs (IRIs)* X, X' such that $m_1(X) \cong m_1(X')$



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

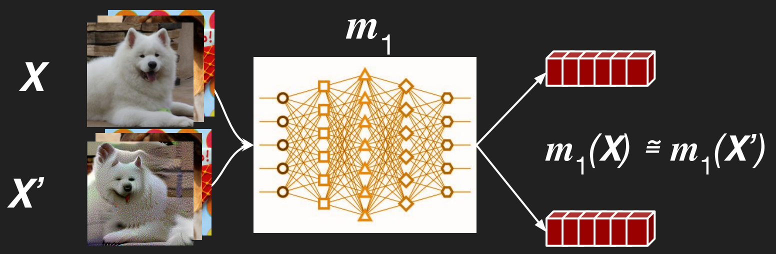
Representation Similarity Measures S_{rep}!



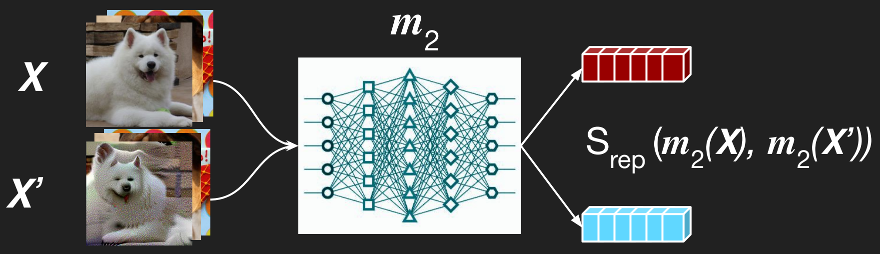
$$S_{inv}(m_2 \mid m_1, X, S_{rep}) = S_{rep}\,(m_2(X), m_2(X'))$$

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 $m_2(X')$



Similarity **T**hrough **I**nverted **R**epresentations

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

(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

ResNet18 Vanilla (m_1),
ResNet18 Vanilla (m_2)

	$m_1 \mid m_2$	$m_2 \mid m_1$
STIR	0.605 $\pm$ 0.013	0.562 $\pm$ 0.023
CKA	0.967 $\pm$ 0.000	

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

ResNet18 Adversarial Training (m_1),
ResNet18 Adversarial Training (m_2)

	$m_1 \mid m_2$	$m_2 \mid m_1$
STIR	0.934 $\pm$ 0.003	0.939 $\pm$ 0.002
CKA	0.937 $\pm$ 0.000	

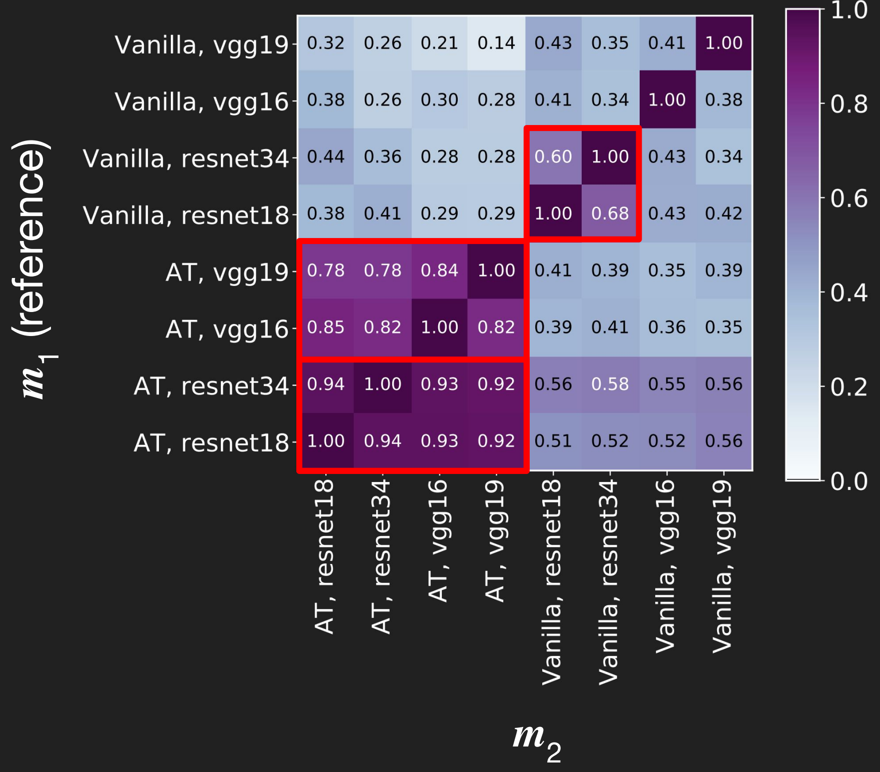
Models trained with Adversarial Training (AT) have **high** levels of shared invariance

Models trained with AT should have higher shared invariance than Vanilla since AT induces invariance to ℓ_p perturbations

CKA is high in both cases and hence does not offer such insights

How do losses & network architectures impact STIR?

Penultimate Layer



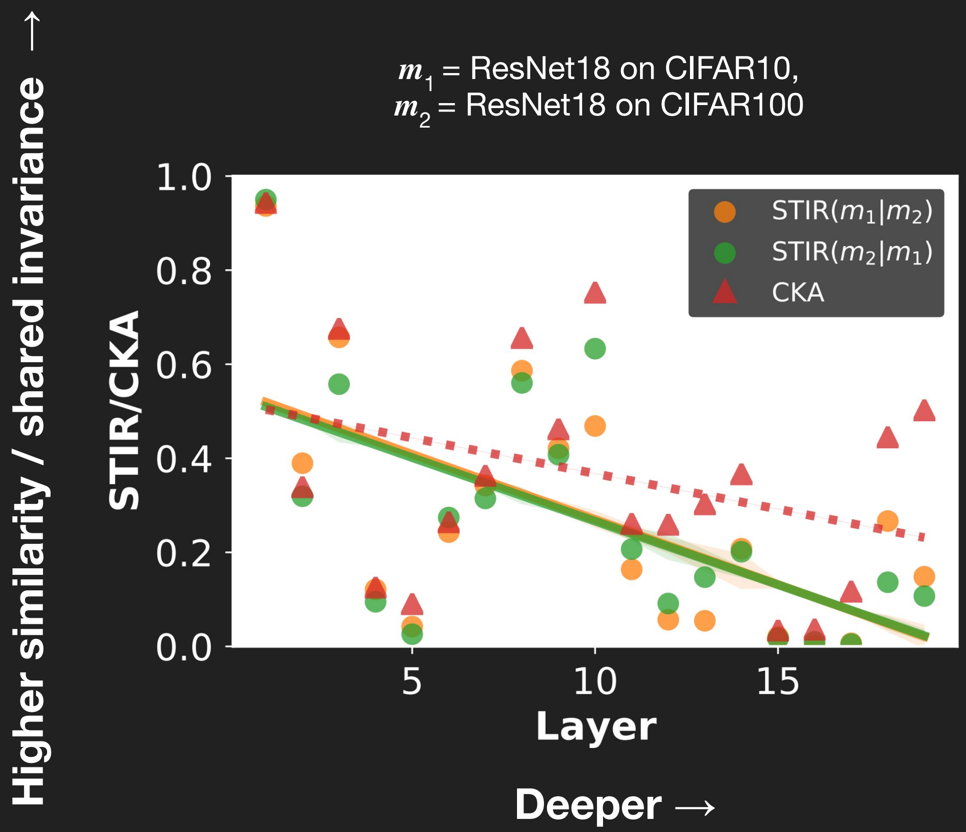
Darker = higher STIR

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 ℓ_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`

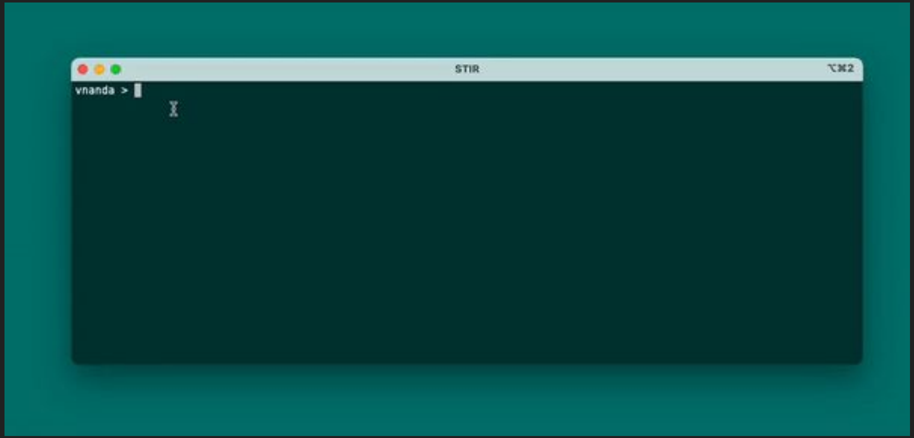
Summary

Proposed **S**imilarity **T**hrough **I**nverted **R**epresentations (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`

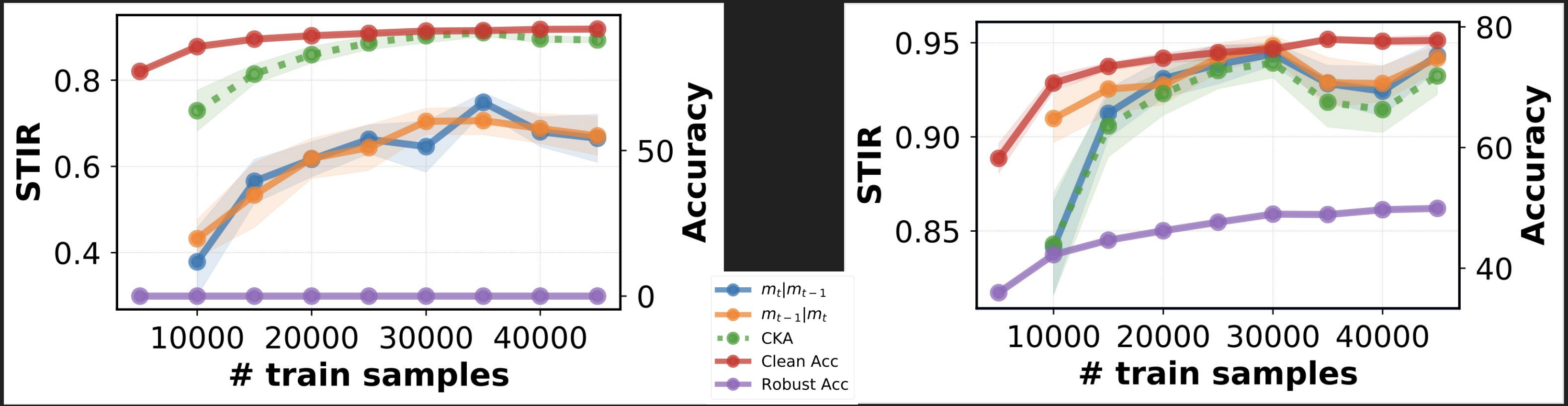
Thank You!

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