Foundational Robustness of *Foundation Models*

NeurIPS 2022 Tutorial

Webpage: bit.ly/neurips-tut-22

Pin-Yu Chen

IBM Research

Sijia Liu

Michigan State University

Sayak Paul

Hugging Face

Outline

1. Basics in foundation model and robustness (Pin-Yu)
2. Foundation models for computer vision (Pin-Yu, Sijia)
3. Foundation models for code (Sijia)
4. Hands-on demo & code walkthroughs (Sayak)
5. Concluding remarks and Q&A (all)

Panel Discussion: Opportunities and Challenges of Robustness in Foundation Models

Panelists: Payel Das (IBM), Alex Gittens (RPI), Celia Cintas (IBM), Bo Li (UIUC), Hildegard Kuehne (Goethe University)

Part 1

Basics in Foundation Model and Robustness

What is *Foundation Model*?

On the Opportunities and Risks of Foundation Models

Rishi Bommasani* Drew A. Hudson Ehsan Adeli Russ Altman Simran Arora
Sydney von Arx Michael S. Bernstein Jeannette Bohg Antoine Bosselut Emma Brunskill
Erik Brynjolfsson Shyamal Buch Dallas Card Rodrigo Castellon Niladri Chatterji
Annie Chen Kathleen Creel Jared Quincy Davis Dorottya Demszky Chris Donahue
Moussa Doumbouya Esin Durmus Stefano Ermon John Etchemendy Kawin Ethayarajh
Li Fei-Fei Chelsea Finn Trevor Gale Lauren Gillespie Karan Goel Noah Goodman
Shelby Grossman Neel Guha Tatsunori Hashimoto Peter Henderson John Hewitt
Daniel E. Ho Jenny Hong Kyle Hsu Jing Huang Thomas Icard Saahil Jain
Dan Jurafsky Pratyusha Kalluri Siddharth Karamcheti Geoff Keeling Fereshte Khani
Omar Khattab Pang Wei Koh Mark Krass Ranjay Krishna Rohith Kuditipudi
Ananya Kumar Faisal Ladhak Mina Lee Tony Lee Jure Leskovec Isabelle Levent
Xiang Lisa Li Xuechen Li Tengyu Ma Ali Malik Christopher D. Manning
Suvir Mirchandani Eric Mitchell Zanele Munyikwa Suraj Nair Avanika Narayan
Deepak Narayanan Ben Newman Allen Nie Juan Carlos Niebles Hamed Nilforoshan
Julian Nyarko Giray Ogut Laurel Orr Isabel Papadimitriou Joon Sung Park Chris Piech
Eva Portelance Christopher Potts Aditi Raghunathan Rob Reich Hongyu Ren
Frieda Rong Yusuf Roohani Camilo Ruiz Jack Ryan Christopher Ré Dorsa Sadigh
Shiori Sagawa Keshav Santhanam Andy Shih Krishnan Srinivasan Alex Tamkin
Rohan Taori Armin W. Thomas Florian Tramèr Rose E. Wang William Wang Bohan Wu
Jiajun Wu Yuhuai Wu Sang Michael Xie Michihiro Yasunaga Jiaxuan You Matei Zaharia
Michael Zhang Tianyi Zhang Xikun Zhang Yuhui Zhang Lucia Zheng Kaitlyn Zhou
Percy Liang*[1]

Center for Research on Foundation Models (CRFM)
Stanford Institute for Human-Centered Artificial Intelligence (HAI)
Stanford University

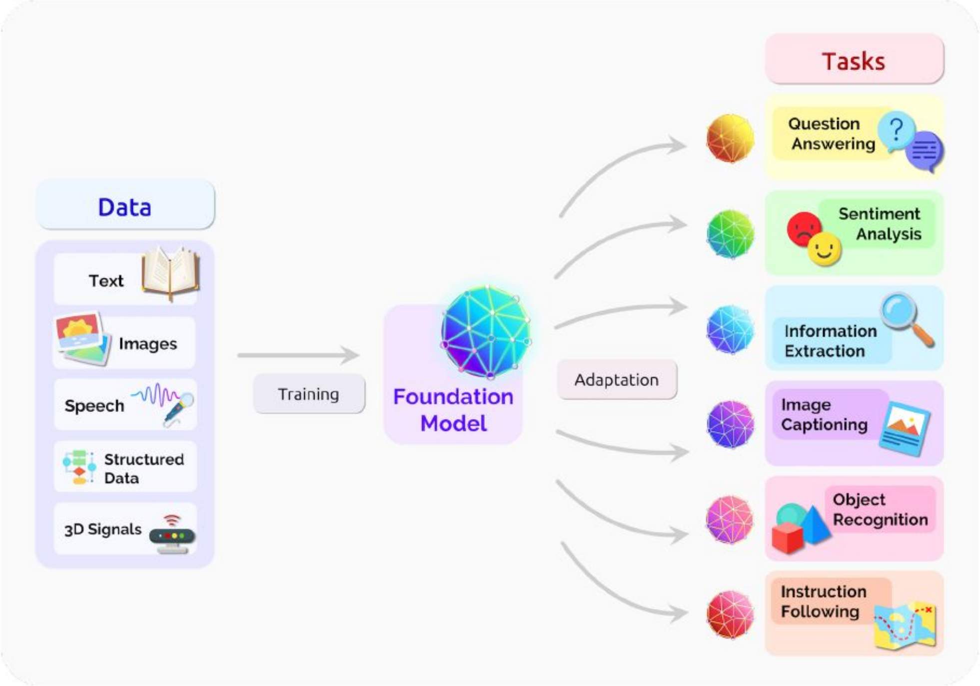


Fig. 2. A foundation model can centralize the information from all the data from various modalities. This one model can then be adapted to a wide range of downstream tasks.

“We introduce the term foundation models to fill a void in describing the paradigm shift we are witnessing... Existing terms (e.g., pretrained model, self-supervised model) partially capture the technical dimension of these models, but fail to capture the significance of the paradigm shift in an accessible manner for those beyond machine learning.

“We also chose the term “foundation" to connote the significance of architectural stability, safety, and security … At present, we emphasize that we do not fully understand the nature or quality of the foundation that foundation models provide; we cannot characterize whether the foundation is trustworthy or not.”

Thomas G. Dietterich @tdietterich

I propose that we adopt the term "Large Self-Supervised Models (LSSMs)" as a replacement for "Foundation Models" and "LLMs". "LLMs" don't capture non-linguistic data and "Foundation Models" is too grandiose. Thoughts? @percyliang

Percy Liang @percyliang · Aug 13
Replying to @tdietterich
The beauty of language is that you can have multiple terms that highlight different aspects of the same object. You don't have to choose. I use "LLM" to talk about LLMs, "self-supervised" for their construction, and "foundation model" for their function. No term can be replaced.

1 2 33

Thomas G. Dietterich @tdietterich · Aug 13
Yes, but as you know, "Foundation" is too close to "Foundational", and many of us find that troubling. That is why I'm proposing a more neutral term. For use, maybe we could just call them "Upstream models".

8 2 55

Show replies

Yann LeCun @ylecun · Aug 13
Though the "large" thing is not going to age well, unless "large" means "larger than what a normal academic lab can train"

7 39

Markus Wulfmeier @markus_with_k · Aug 17
Is early work in deep learning still 'deep'? There's a good argument to be made that the terminology still applies.

(Not that I haven't had papers partially rejected because the deep network didn't have enough layers to be called deep 😅)

Hilde Kuehne @HildeKuehne · 22h
Replying to @tdietterich and @percyliang
I would vote for calling it pretrained backbones (PB) as we always did. If you want to make it large, call it LPB... just a reasonable description of reality with a bit less magic and glitter

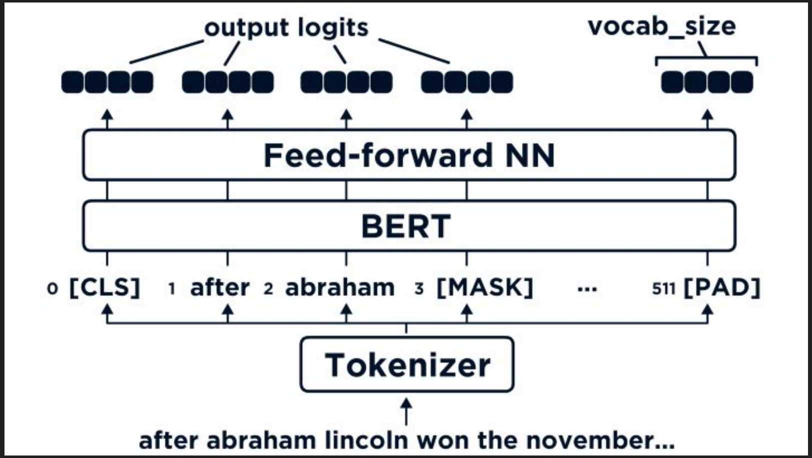
Our take on Foundation Model:
A machine learning paradigm featuring ***task-agnostic pre-training*** and ***task-specific fine-tuning*** via neural networks

- Task-agnostic pre-training (with unlabeled/noisy data)
  - Self-supervised learning of data representations
    - Supervision-free pre-training
      - Data scalability (Texts, Images, Speech, etc)
    - Use of auxiliary tasks
      - Masked prediction (of tokens)
      - Contrastive learning
  - Generic representation learning of a data modality (aka a data encoder)
- Task-specific fine-tuning (with labeled data)
  - Linear probing (training a linear head on representations)
  - Full fine-tuning (training both the linear head and the encoder)
- Examples:
  - Large language models such as GPT-3 and BLOOM
  - Transformer-based neural networks for different data modalities

*We do not exclude the use of supervised pre-training for foundation models

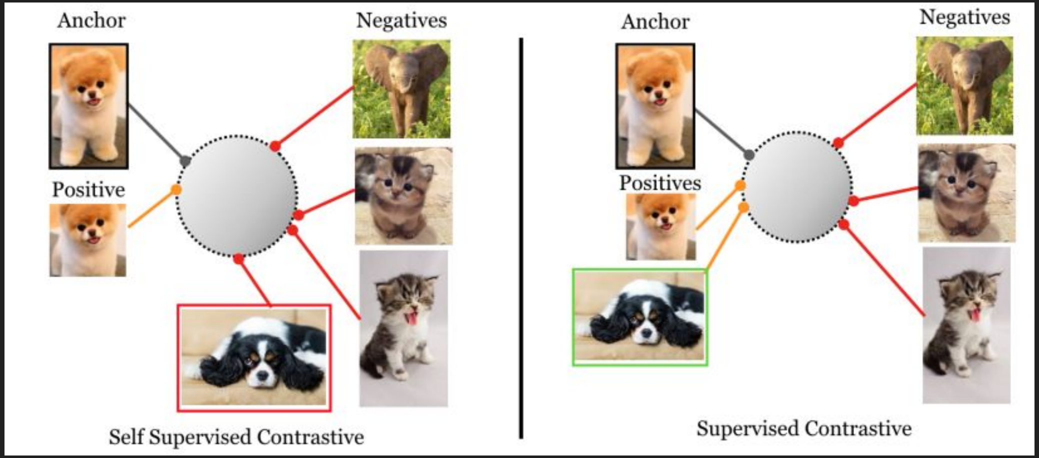
Examples of Task-agnostic Pre-training

Masked prediction



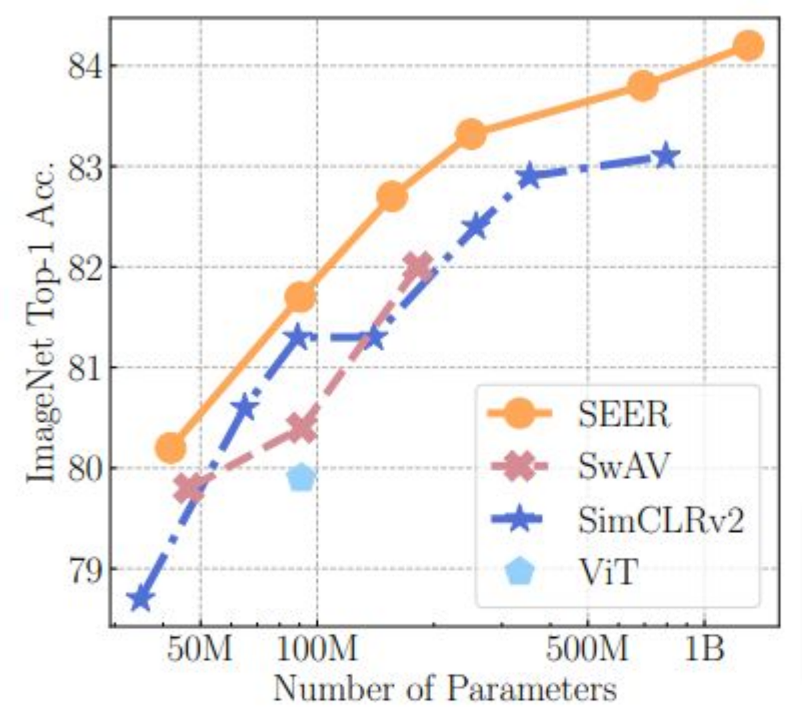
James Briggs. Masked-Language Modeling With BERT. TowardsDataScience, 2021

Contrastive learning



Khosla et al. Supervised Contrastive Learning. NeurIPS 2020

Data & Model Scalability with Self-Supervised Learning



Self-supervised Pretraining of Visual Features in the Wild

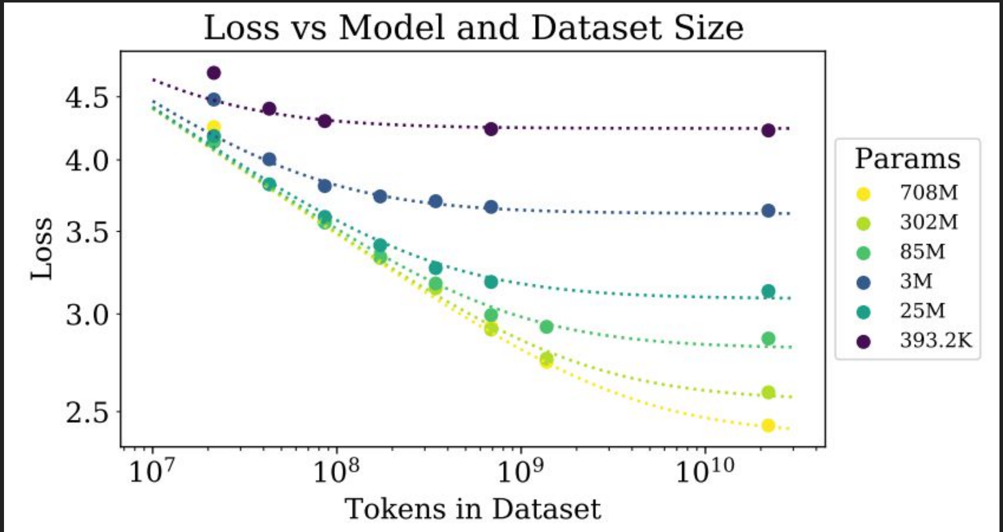
Priya Goyal[1] Mathilde Caron[1,2] Benjamin Lefaudeux[1] Min Xu[1] Pengchao Wang[1] Vivek Pai[1]
Mannat Singh[1] Vitaliy Liptchinsky[1] Ishan Misra[1] Armand Joulin[1] Piotr Bojanowski[1]

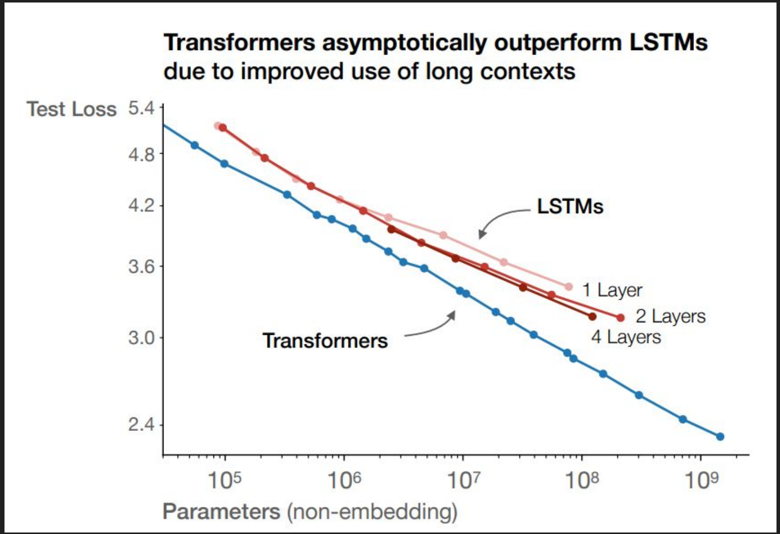
[1] Facebook AI Research [2] Inria*

Code: https://github.com/facebookresearch/vissl

"Our final SElf-supERvised (SEER) model, a RegNetY with 1.3B parameters trained on 1B random images with 512 GPUs achieves 84.2% top-1 accuracy, surpassing the best self-supervised pretrained model by 1% and confirming that self-supervised learning works in a real world setting."

Neural Scaling Laws





Kaplan et al. Scaling Laws for Neural Language Models. Arxiv 2020

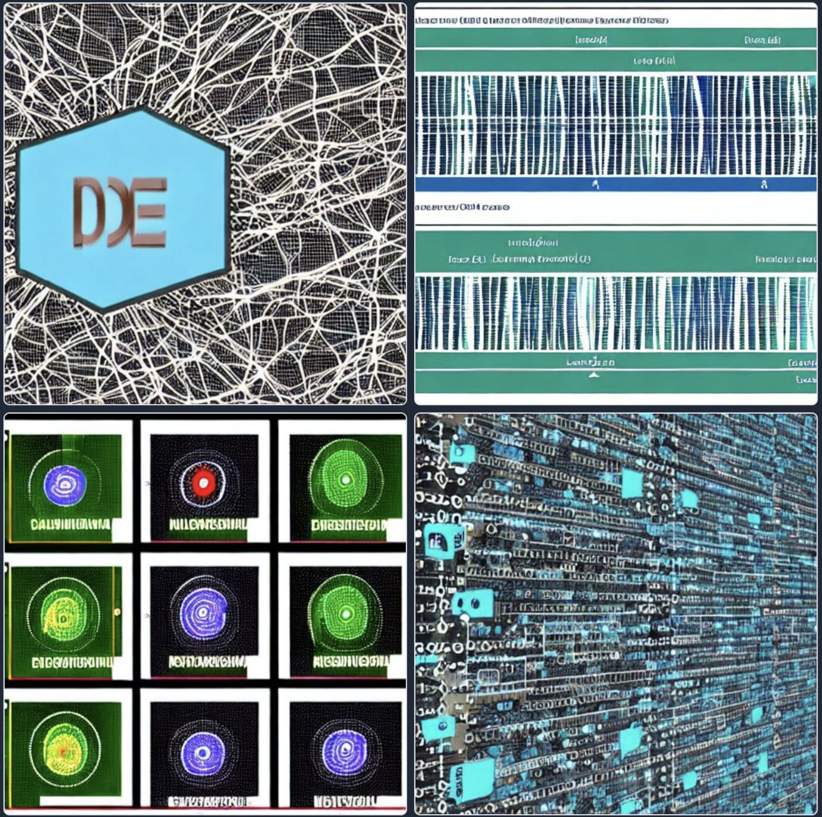
What is *Foundational Robustness*?

Evaluation and enhancement of (and sometimes certifiable) model correctness against natural and adversarial data shifts

Stable Diffusion Demo

Stable Diffusion is a state of the art text-to-image model that generates images from text.
For faster generation and forthcoming API access you can try DreamStudio Beta

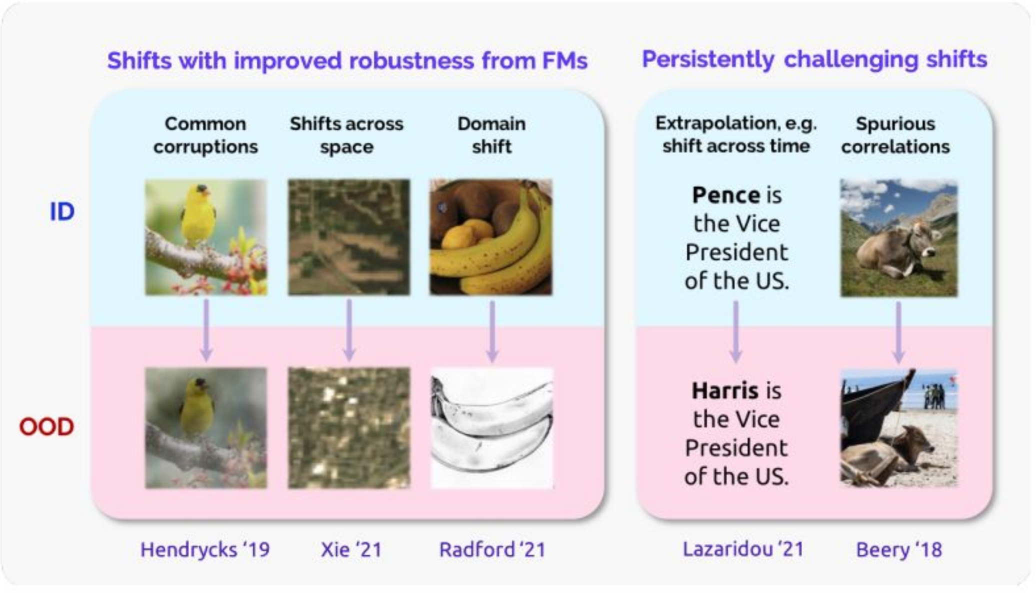
Robustness of Deep Learning models. **Generate image**



On the Opportunities and Risks of Foundation Models 109

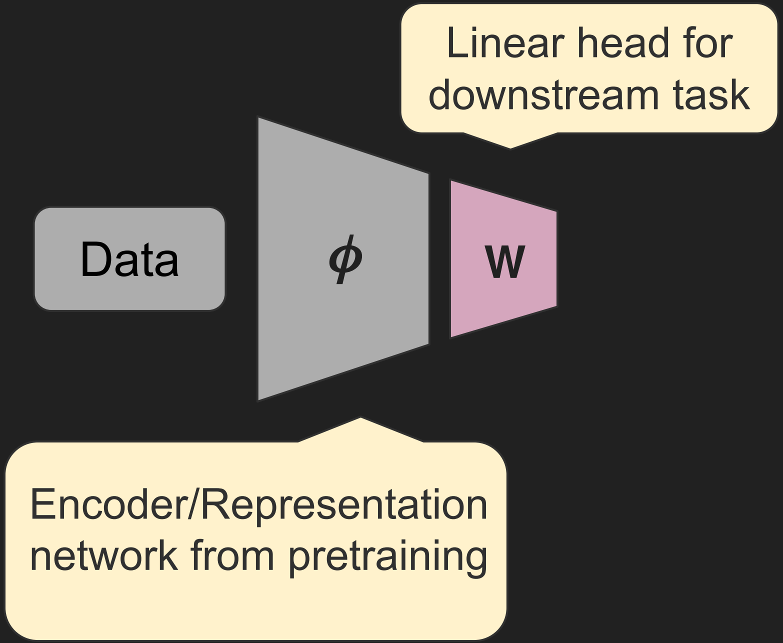
4.8 Robustness to distribution shifts

Authors: Sang Michael Xie, Ananya Kumar, Rohan Taori, Tony Lee, Shiori Sagawa, Pang Wei Koh, Tatsunori Hashimoto



https://huggingface.co/spaces/stabilityai/stable-diffusion

Formalizing Robustness of Foundation Models (1)



$\theta=\{\phi, W\}$

Pre-training on ϕ

Fine-tuning principles:

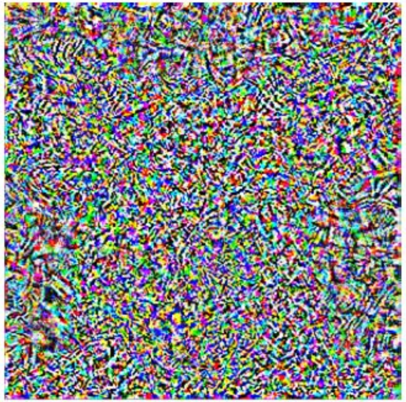
- *Standard linear probing*:
  Fix ϕ, train W
- *Full fine-tuning*:
  Train both ϕ and W

ML Predictions Are (Mostly) Accurate but Brittle

"pig" (91%)

+ 0.005 x

noise (NOT random)



=

"airliner" (99%)

J. Z. Kolter and A. Madry: Adversarial Robustness - Theory and Practice (NeurIPS 2018 Tutorial)

How to measure the quality of representations from foundation models?

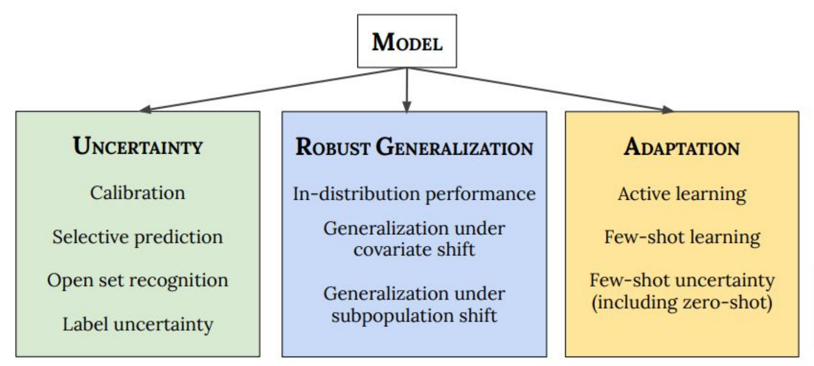
Benchmarking Representation Robustness and Beyond (1)

Diverse test sets using real downstream data (task-specific)

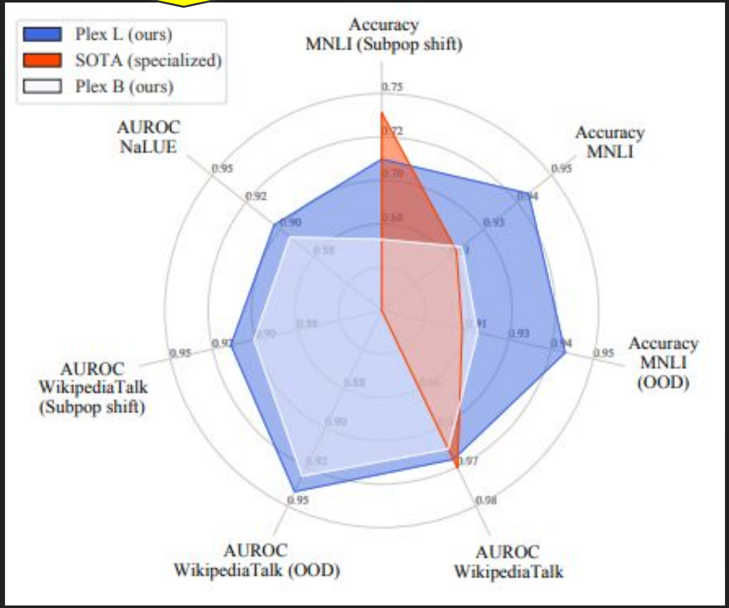
PLEX: Towards Reliability Using Pretrained Large Model Extensions

Dustin Tran[*1], Jeremiah Liu[1], Michael W. Dusenberry[1], Du Phan[1], Mark Collier[1], Jie Ren[1], Kehang Han[1], Zi Wang[1], Zelda Mariet[1], Huiyi Hu[1], Neil Band[2], Tim G. J. Rudner[2], Karan Singhal[1], Zachary Nado[1], Joost van Amersfoort[2], Andreas Kirsch[2], Rodolphe Jenatton[1], Nithum Thain[1], Honglin Yuan[1†], Kelly Buchanan[1†], Kevin Murphy[1], D. Sculley[1], Yarin Gal[2], Zoubin Ghahramani[1], Jasper Snoek[1], Balaji Lakshminarayanan[1]

[1]*Google* [2]*University of Oxford*

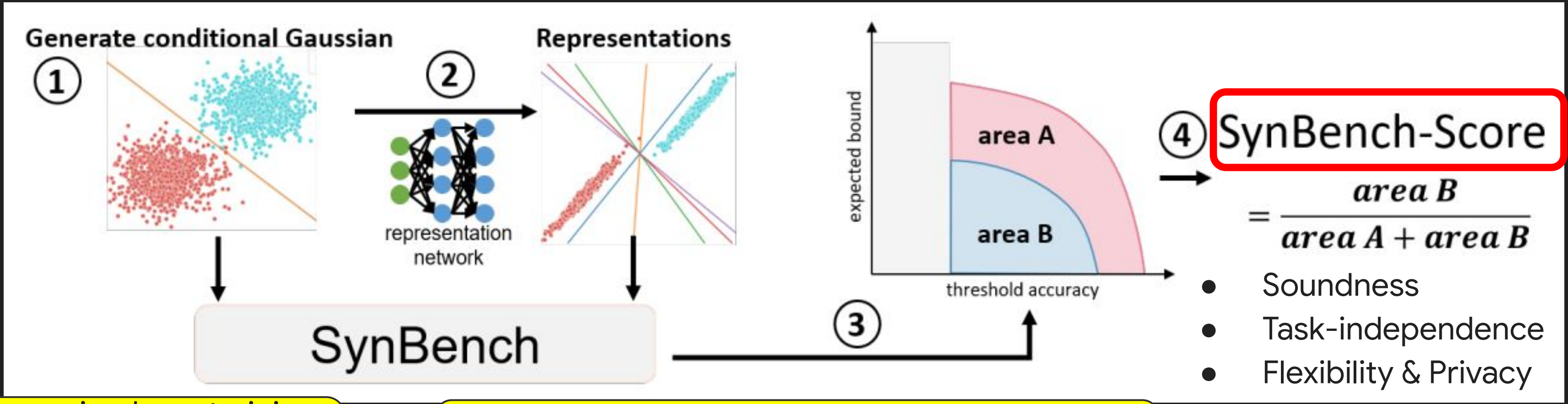


"We devise 10 types of tasks over 40 datasets in order to evaluate different aspects of reliability on both vision and language domains."



Benchmarking Representation Robustness and Beyond (2)

Synthetic data with ideal reference (task-agnostic)



Supervised pre-training on ImageNet-21K

Linear heads with different robustness margins

$a_t = 0.7$	$\epsilon = 0$	$\epsilon = 0.2$	$\epsilon = 0.4$	$\epsilon = 0.6$	$\epsilon = 0.8$	CIFAR10	CIFAR10-c
ViT-B/16 + ImageNet-1K	0.33	0.37	0.32	0.20	0.06	95.0	81.2
ViT-B/16	0.20	0.23	0.18	0.07	0.01	89.6	71.4

Ching-Yun Ko, Pin-Yu Chen, Jeet Mohapatra, Payel Das, and Luca Daniel. SynBench: Task-Agnostic Benchmarking of Pretrained Representations using Synthetic Data. arxiv 2022

Other Aspects of Trustworthiness in Foundation Models (no covered in this tutorial)

Language Models are Few-Shot Learners

Tom B. Brown* Benjamin Mann* Nick Ryder* Melanie Subbiah*
Jared Kaplan† Prafulla Dhariwal Arvind Neelakantan Pranav Shyam Girish Sastry
Amanda Askell Sandhini Agarwal Ariel Herbert-Voss Gretchen Krueger Tom Henighan
Rewon Child Aditya Ramesh Daniel M. Ziegler Jeffrey Wu Clemens Winter
Christopher Hesse Mark Chen Eric Sigler Mateusz Litwin Scott Gray
Benjamin Chess Jack Clark Christopher Berner
Sam McCandlish Alec Radford Ilya Sutskever Dario Amodei

OpenAI

NeurIPS 2020

- 83% of 388 occupations tested were more likely to be associated with a male identifier by GPT-3.

- “Black” had a consistently low sentiment.

Extracting Training Data from Large Language Models

USENIX 2021

Nicholas Carlini[1] Florian Tramèr[2] Eric Wallace[3] Matthew Jagielski[4]
Ariel Herbert-Voss[5,6] Katherine Lee[1] Adam Roberts[1] Tom Brown[5]
Dawn Song[3] Úlfar Erlingsson[7] Alina Oprea[4] Colin Raffel[1]

[1]*Google* [2]*Stanford* [3]*UC Berkeley* [4]*Northeastern University* [5]*OpenAI* [6]*Harvard* [7]*Apple*

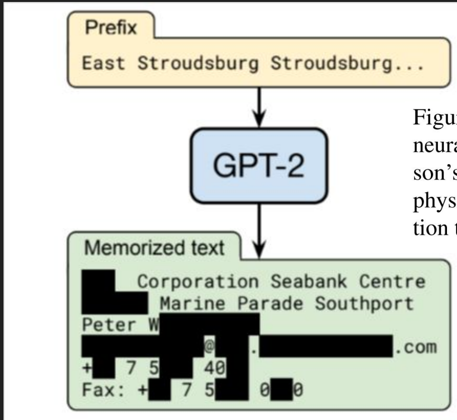


Figure 1: **Our extraction attack.** Given query access to a neural network language model, we extract an individual person’s name, email address, phone number, fax number, and physical address. The example in this figure shows information that is all accurate so we redact it to protect privacy.

*Many broader topics were discussed in “On the Opportunities and Risks of Foundation Models”

Part 2

Foundation Models for Computer Vision

Robustness Evaluation & Attribution of Vision Transformers

Sayak Paul* and Pin-Yu Chen*. Vision transformers are robust learners. AAAI 2022
Rulin Shao, Zhouxing Shi, Jinfeng Yi, Pin-Yu Chen, and Cho-Jui Hsieh. On the adversarial robustness of vision transformers. TMLR 2022

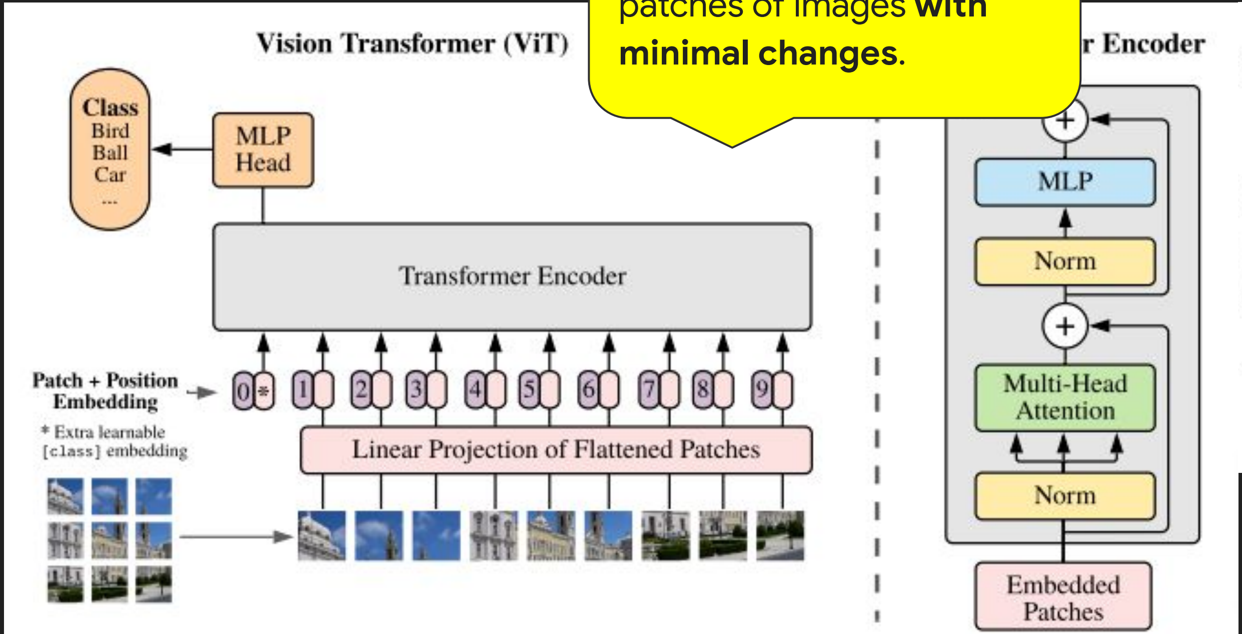
Vision Transformers

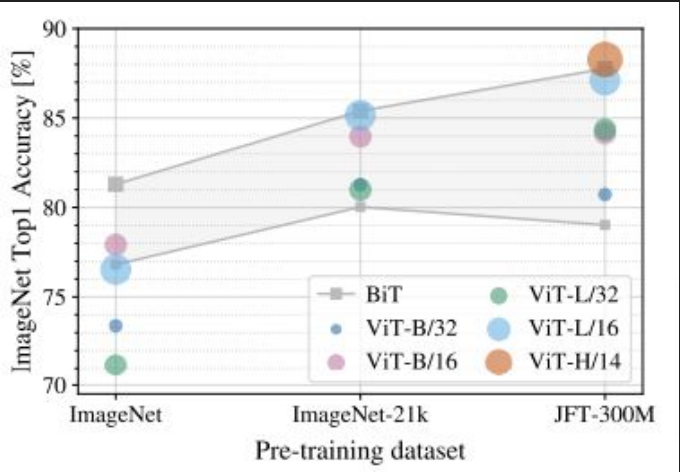
AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

Alexey Dosovitskiy*,†, Lucas Beyer*, Alexander Kolesnikov*, Dirk Weissenborn*, Xiaohua Zhai*, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby*,†
*equal technical contribution, †equal advising
Google Research, Brain Team
{adosovitskiy, neilhoulsby}@google.com

ICLR 2021

Pure Transformer (Vaswani et al., NeurIPS'17) applied to patches of images **with minimal changes**.





How about robustness?

Robustness Evaluation

Out-of-distribution Generalization

Dataset	Purpose
ImageNet-C (Hendrycks and Dietterich 2019)	Common corruptions
ImageNet-P (Hendrycks and Dietterich 2019)	Common perturbations
ImageNet-R (Hendrycks et al. 2020)	Semantic shifts
ImageNet-O (Hendrycks et al. 2021)	Out-of-domain distribution
ImageNet-A (Hendrycks et al. 2021)	Natural adversarial examples
ImageNet-9 (Xiao et al. 2021)	Background dependence

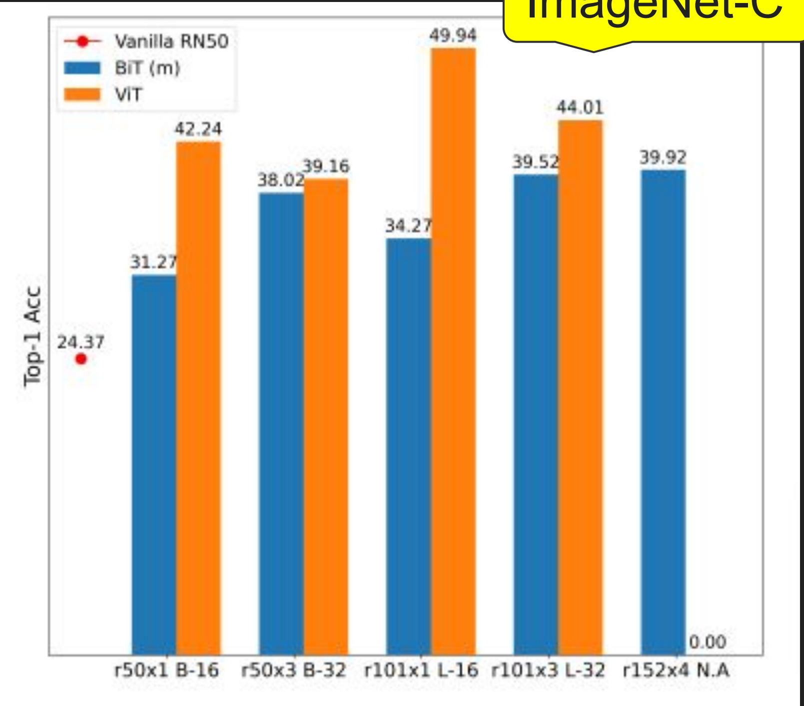
Robustness to Adversarial Perturbations

- Empirical robustness
  - Minimize$_{\boldsymbol{\delta} \in S}$ loss$_{attack}$(x+$\boldsymbol{\delta}|\boldsymbol{\theta}$), where S is a neighborhood of x.
  - Example: loss$_{attack}$ = negative cross entropy of $\mathbf{f}_{\boldsymbol{\theta}}$(x) and y
- Certified robustness
  - Find a neighborhood R around x such that $\mathbf{f}_{\boldsymbol{\theta}}$(x) = $\mathbf{f}_{\boldsymbol{\theta}}$(x') for any x'∈R
  - Example: randomized smoothing

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. ICML 2019

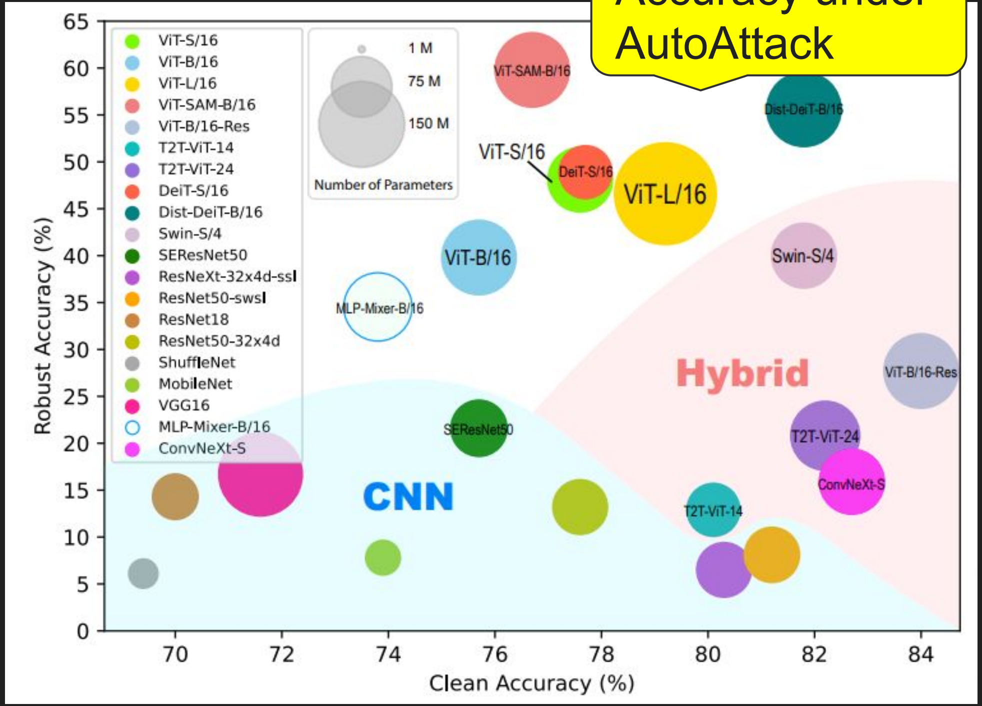
Intrinsic Robustness in Vision Transformers (ViTs)

ImageNet-C



Models of different sizes
BiT: Big Transfer (CNN); RN: ResNet

Accuracy under AutoAttack



Francesco Croce and Matthias Hein. [Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks](). ICML 2020

More Key Findings

- ViTs outperform others on most but not all OOD benchmarks [AAAI'22]
- Pure ViTs possess better certified robust accuracy than CNNs [TMLR'22]
- Modern CNN design helps bridge the performance gap between CNNs and ViTs (e.g., ConvNeXt, MLP-Mixer, SEResNet) [TMLR'22]
- (Standard) pre-training helps OOD robustness but not necessarily adversarial robustness [AAAI'22, TMLR'22]

ImageNet-9: detecting vulnerable image foregrounds

Model	Challenge Accuracy (%)
BiT-m r101x3	3.78
ViT L-16	20.02
ResNet-50	22.3

ViT B/16

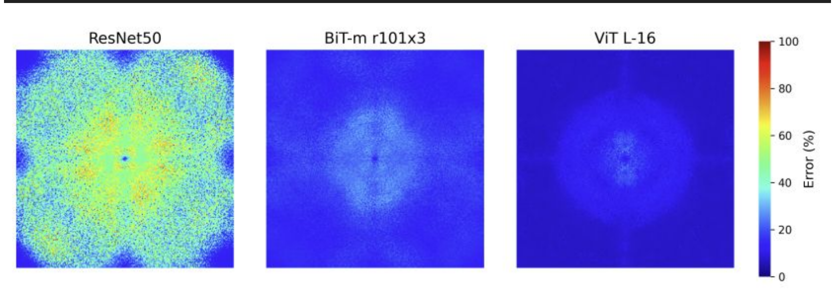
Pre-training	ImageNet-A (Top-1 Acc)	ImageNet-R (Top-1 Acc)	ImageNet-O (AUPR)
ImageNet-1k	8.630994	28.213835	26.25
ImageNet-21k	21.746947	41.815233	54.61

Robustness Attribution for ViTs

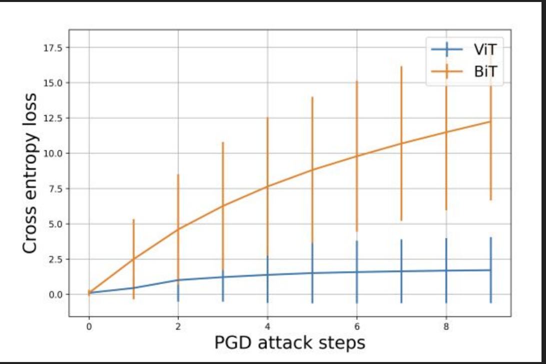
Better use of global context

Masking Factor	Top-1 Acc (BiT)	Top-1 Acc (ViT)
0	79	83
0.05	76	82.3
0.1	75	81.4
0.2	72.4	77.9
0.5	52	60.4

Lower model sensitivity



Smoother loss landscape



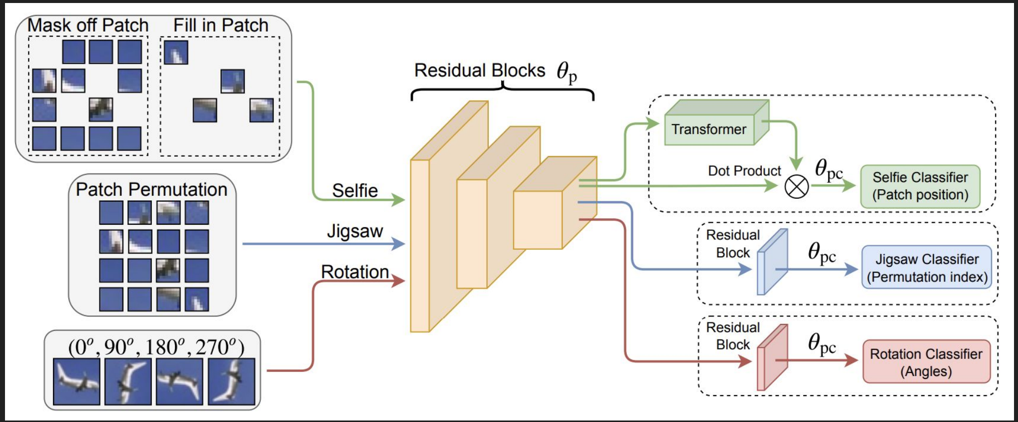
*More results can be found in "[Vision Transformers are Robust Learners]"

Dong Yin, Raphael Gontijo Lopes, Jon Shlens, Ekin Dogus Cubuk, and Justin Gilmer.
A fourier perspective on model robustness in computer vision. NeurIPS 2019

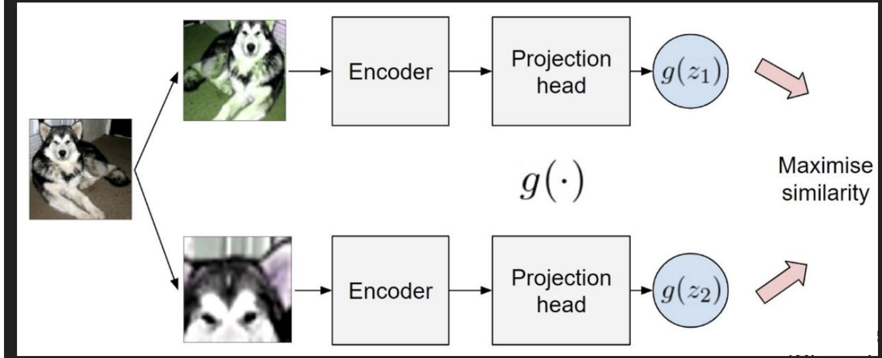
(Adversarial) Robustness Transfer: From (Self-supervised) Pre-training to Fine-tuning

Robust Self-supervised Pre-training

- Given a **robust** pre-trained model, is it possible to transfer robustness to downstream tasks?
- **Self-supervised pre-training:** Rotation prediction [Gidaris et al., 2018], Jigsaw [Noroozi et al., 2017], Selfie [Trinh et al., 2019], SimCLR [Chen et al., 2020]



[Chen et al., 2020]



(SimCLR architecture; image from TDS blog)

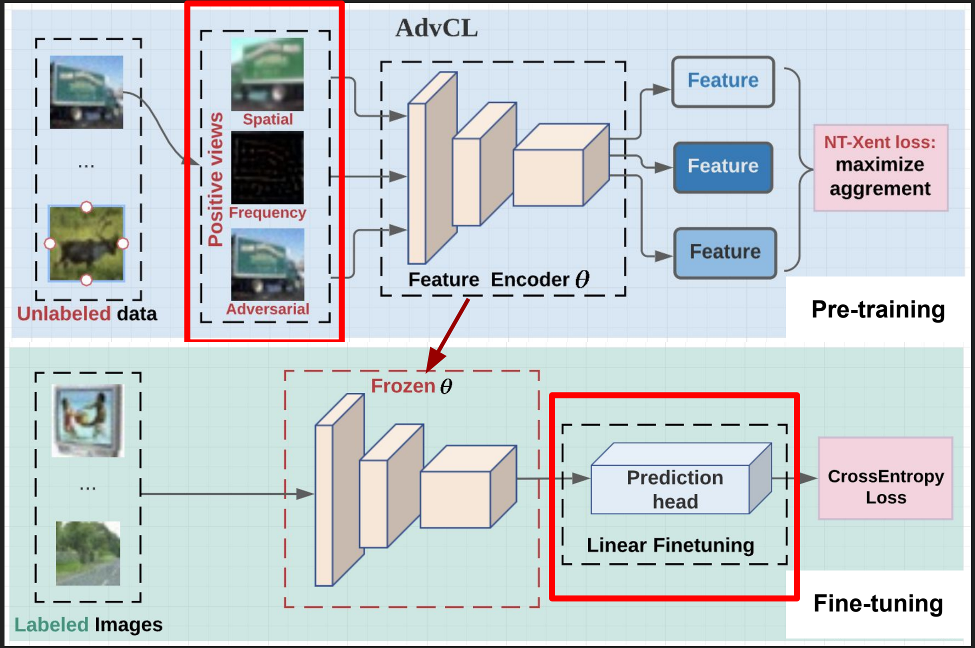
Robust Self-supervised Pre-training

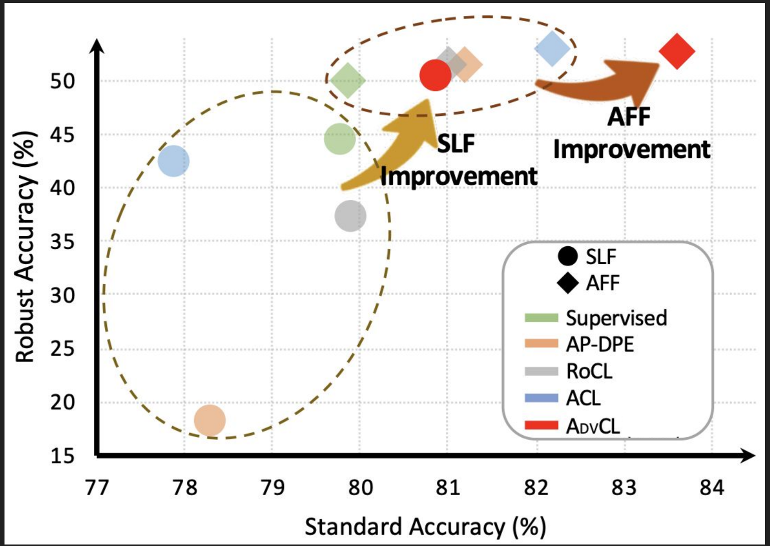
- **Challenge:** In most of robust self-supervised pre-training mechanisms, robustness is **difficult to transfer** to downstream fine-tuning tasks **unless** robust fine-tuning is also performed [Chen et al., 2020]

- **Solutions** to improving robustness transfer from pre-training to fine-tuning:
    - Adversarial contrastive learning [Fan et al., 2021, Gowal et al., 2021]
    - Robust pre-training + model sparsification [Chen et al., 2022]

Adversarial Contrastive Learning (AdvCL)

- **AdvCL** [Fan et al., 2021]: Leverages **adversary-related data transformations** (i.e., 'views') to create **'positive'** data pairs

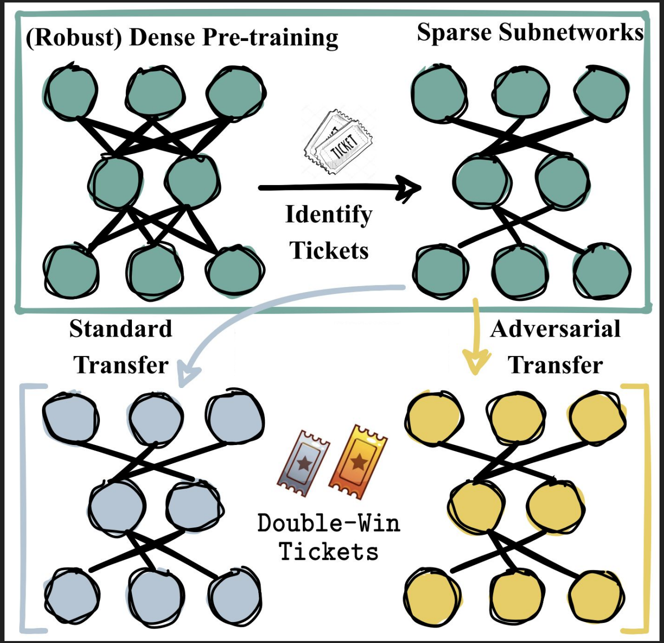




SLF: Standard Linear Fine-tuning
AFF: Adversarial Full Fine-tuning

Robust Pre-training + Model Pruning

- **Model Pruning:** Finding **sparse subnetwork** from dense model without performance loss — '**Winning ticket**' in lottery ticket hypothesis (LTH) [Frankle et al., 2018]



[Chen et al., 2022]

- Sparsity from pre-trained robust model can be transferred on diverse downstream tasks, to preserve BOTH standard and robust generalization, under BOTH standard and adversarial training regimes

Part 3

Foundation Models for Code

Emerging AI Applications to Code/Programming Language

2017

A Survey of Machine Learning for Big Code and Naturalness

MILTIADIS ALLAMANIS, Microsoft Research
EARL T. BARR, University College London
PREMKUMAR DEVANBU, University of California, Davis
CHARLES SUTTON, University of Edinburgh and The Alan Turing Institute

Project CodeNet: A Large-Scale AI for Code Dataset for Learning a Diversity of Coding Tasks

Ruchir Puri, David S. Kung, Geert Janssen, Wei Zhang, Giacomo Domeniconi, Vladmir Zolotov, Julian Dolby, Jie Chen, Mihir Choudhury, Lindsey Decker, Veronika Thost, Luca Buratti, Saurabh Pujar, Ulrich Finkler. 2021

[GitHub]

dataset

NeurIPS’21

Proof Engineering, Adaptation, Repair, and Learning for Software (PEARLS)

DARPA AIE 2021

INACTIVE

Contract Opportunity

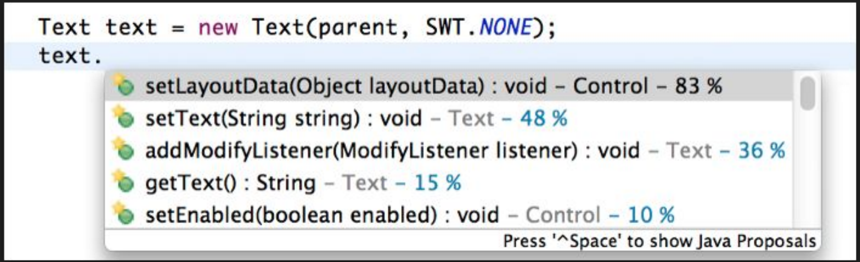
Notice ID
DARPA-PA-21-04-04

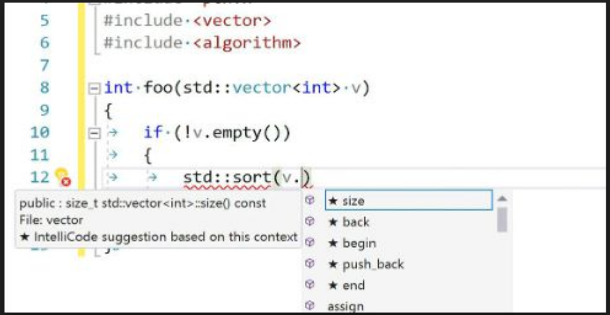
Related Notice

Department/Ind. Agency
DEPT OF DEFENSE
Sub-tier
DEFENSE ADVANCED RESEARCH PROJECTS AGENCY (DARPA)
Office
DEF ADVANCED RESEARCH PROJECTS AGCY

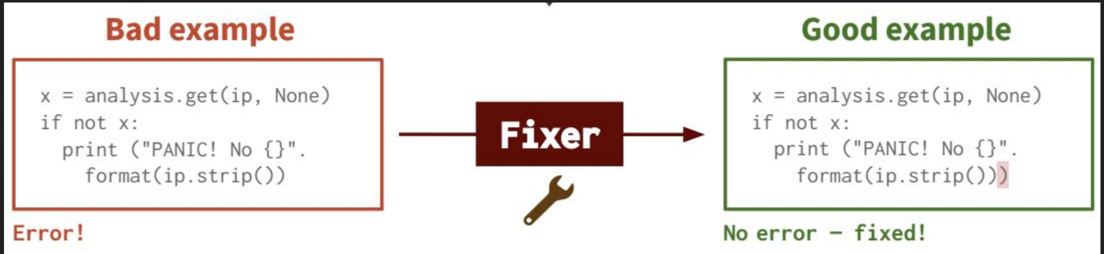
Emerging AI Applications to Code

- Autocompletion [Svyatkovskiy et al., 2021]

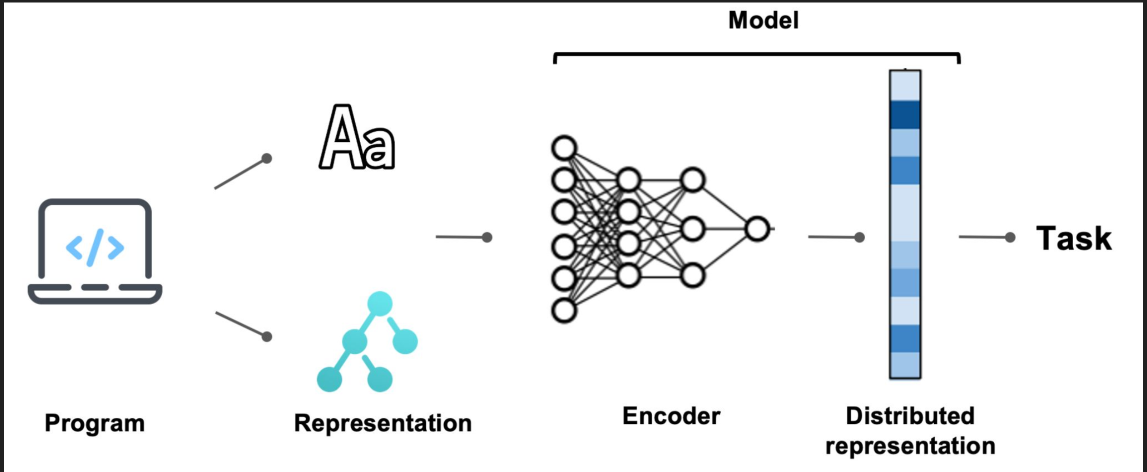




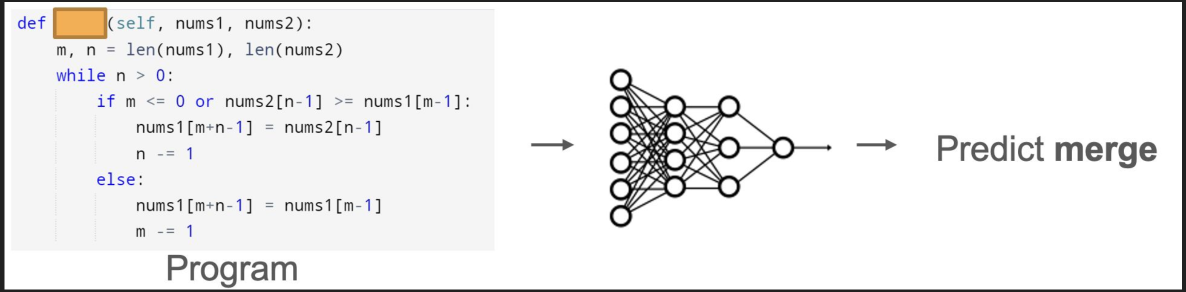
- Code repair [Yasunaga et al., 2021]



ML Model for Code Tasks



Example: Code summarization task [Allamanis et al., 2016]

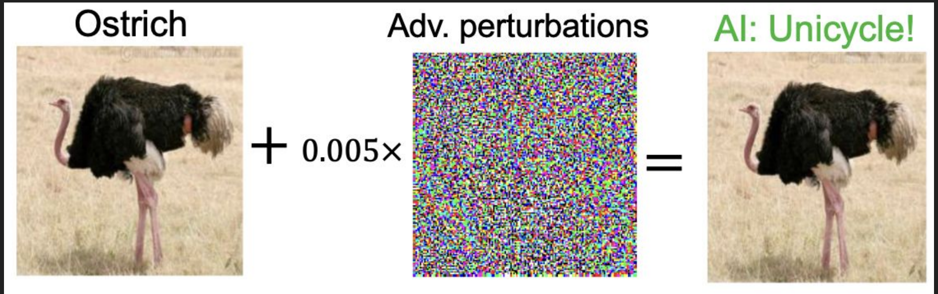


(Worst-case) Robustness Problem of Code Model?

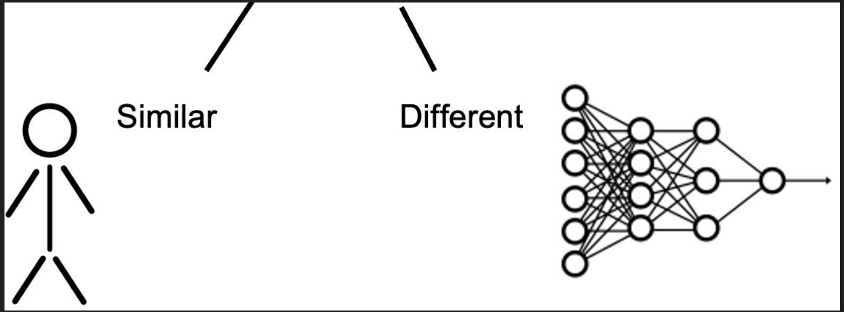
Evaluation: “Perturb” an input program (*P)* to justify robustness of code model

Challenge: How to define “code perturbation”?

Image:



Visual similarity:



(Worst-case) Robustness Problem of Code Model?

Evaluation: “Perturb” an input program (*P)* to justify robustness of code model?

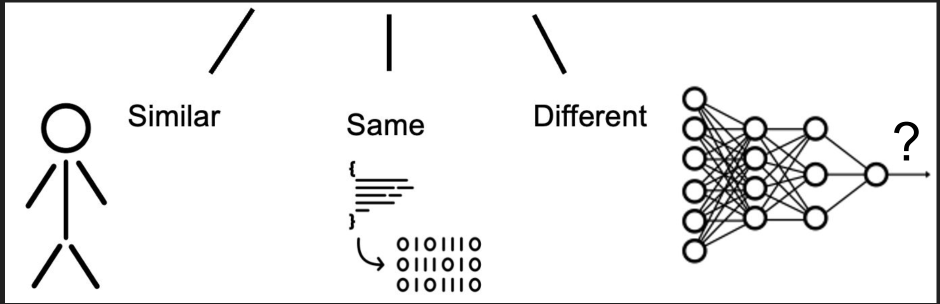
Challenge: How to define “code perturbation”?

Program (P):

```
def foo():
    x = foo1()
    y = foo2()
    print("Hello World")
```

+ ? = ?

Functional similarity:



Obfuscation as Perturbation Operation in Code

```python
def foo():
    x = foo1()
    y = foo2()
    print("Hello World")
```

Replacing x with Q:

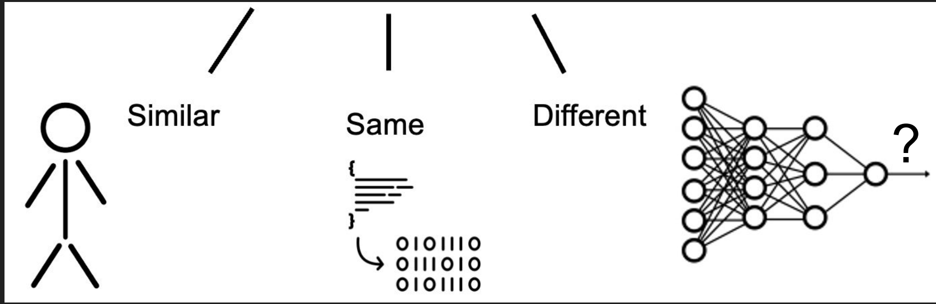
+ Q =

```python
def foo():
    Q = foo1()
    y = foo2()
    print("Hello World")
```

Original program

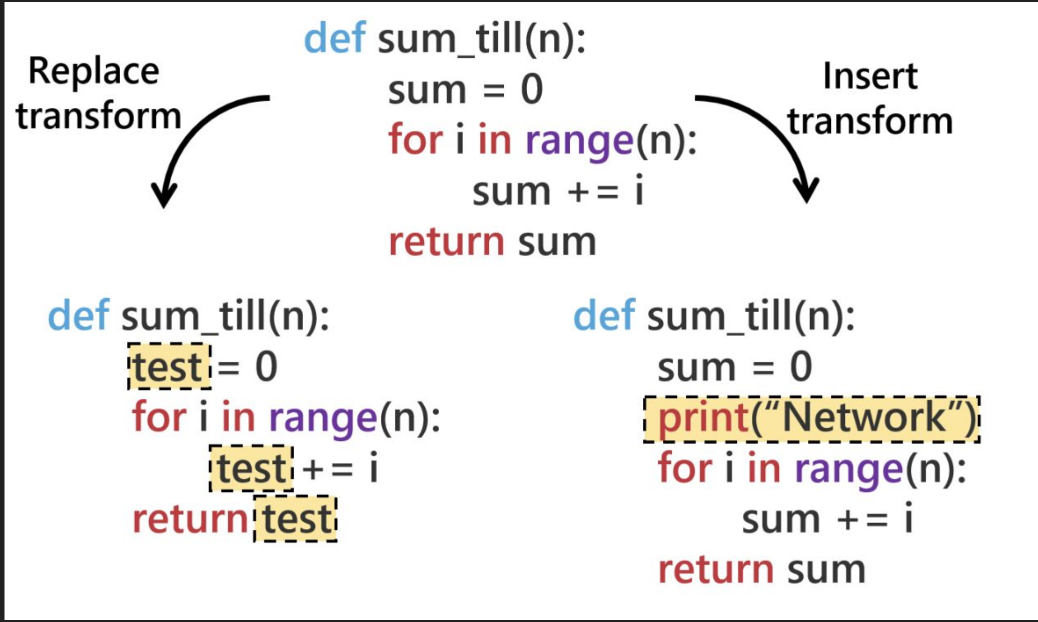
Obfuscation: Variable renaming/replacement

Obfuscated program



Obfuscation as Perturbation Operation in Code

Obfuscation: Two broad classes – replace and insert transformations



Obfuscation as Perturbation Operation in Code

Original program (non-adversarial)

```
def __setitem__(self, name, val):
    name, val = forbid_multi_line_headers(name, val, self.encoding)
    MIMEText.__setitem__(self, name, val)
```

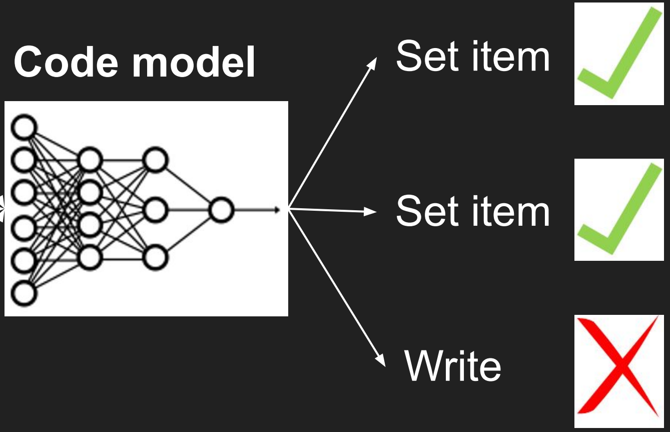
Obfuscated program (non-adversarial)

```
def __setitem__(self, name, val):
    virtualname, val = forbid_multi_line_headers(name, val, self.encoding)
    MIMEText.__setitem__(self, virtualname, val)
```

Adversarial program

```
def __setitem__(self, qisrc, val):
    name, val = forbid_multi_line_headers(qisrc, val, self.encoding)
    MIMEText.__setitem__(self, name, val)
```

Code model



Adversarial Program for Robustness Evaluation of Code Models

Adversarial program: Optimized obfuscated code to fool code models

Two design problems:

- Site selection: Where to perturb in the code?
- Perturbation content: How to perturb?

Solution: First-order optimization-based adversarial program generation methods [Yefet et al., 2020] [Ramakrishnan et al., 2020] [Srikant et al., 2021]

Adversarial Program Generation

```python
def foo():
    x = foo1()
    y = foo2()
    print("Hello World")
```

P = x y foo1 foo2

z is our **site selection variable**
u is the **site perturbation variable**

```python
def foo():
    x = foo1()
    y = foo2()
    print("Hello World")
```

z

u

{ x, y, foo1, foo2, T, Q, R, a, b, c }

```python
def foo():
    ▭ = foo1()
    y = ▭
    print("Hello World")
```

z

u

{ x, y, foo1, foo2, T, Q, R, a, b, c }

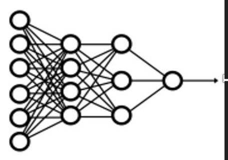
P =	x	y	foo1	foo2
z =	1	0	0	1
u =	0 0 0 .. 1 .. 0			0 1 0 .. 0 .. 0

Optimization for site selection and perturbation

$$P_{perturbed} = \mathbf{z} \,.\, \mathbf{u} + (1 - \mathbf{z}) \,.\, P$$

where $\sum \mathbf{z} \leq k, \quad \mathbf{z} \in [0, 1]^n$

$\sum \mathbf{u_i} = 1, \quad \mathbf{u_i} \in [0, 1]^{|V|}$



Target label

Example of Adversarial Program for Code Summarization

Original program

Unperturbed

```
def __call__(self, *a, **ka):
  for key, value in dict(*a, **ka).items(): setattr(self, key, value)
  return self
```

Prediction: call

Random site selection

Random site-selection; Optimal site-perturbation (Ramakrishnan et al., 2020)

```
def __call__(self, *a, **ka):
  for save, value in dict(*a, **ka).items(): setattr(self, save, value)
  return self
```

Prediction: call

Optimized site selection

Optimal site-selection; Optimal site-perturbation + Smoothing

```
def __call__(self, *datetime, **ka):
  for key, value in dict(*datetime, **ka).items(): setattr(self, key, value)
  return self
```

Prediction: create

Example from [Srikant et al., 2021]

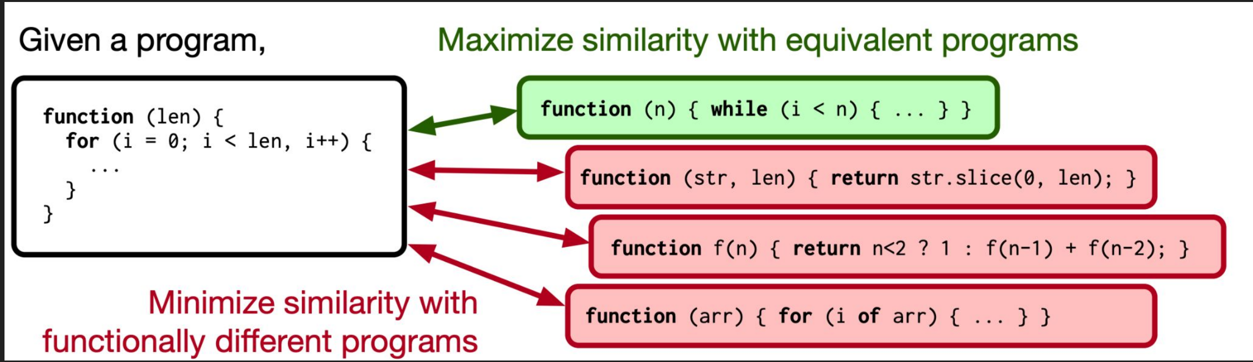
Takeaways – Robustness Evaluation of Code Models

- Code obfuscation is a natural way to define code ‘perturbation’
- There exists worst-case obfuscation that can transform ‘benign’ code to ‘adversarial’ code for ML models
- In design of adversarial code, both ‘where to perturb’ and ‘how to perturb’ matter

How to Robustify Code Models?

Contrastive representation learning for code: Since 'perturbation' (obfuscation) is a type of code transformation, leverages contrastive learning to learn 'invariant' code representations across 'diverse' transformations

E.g., ContraCode (Jain, et al., 2022)

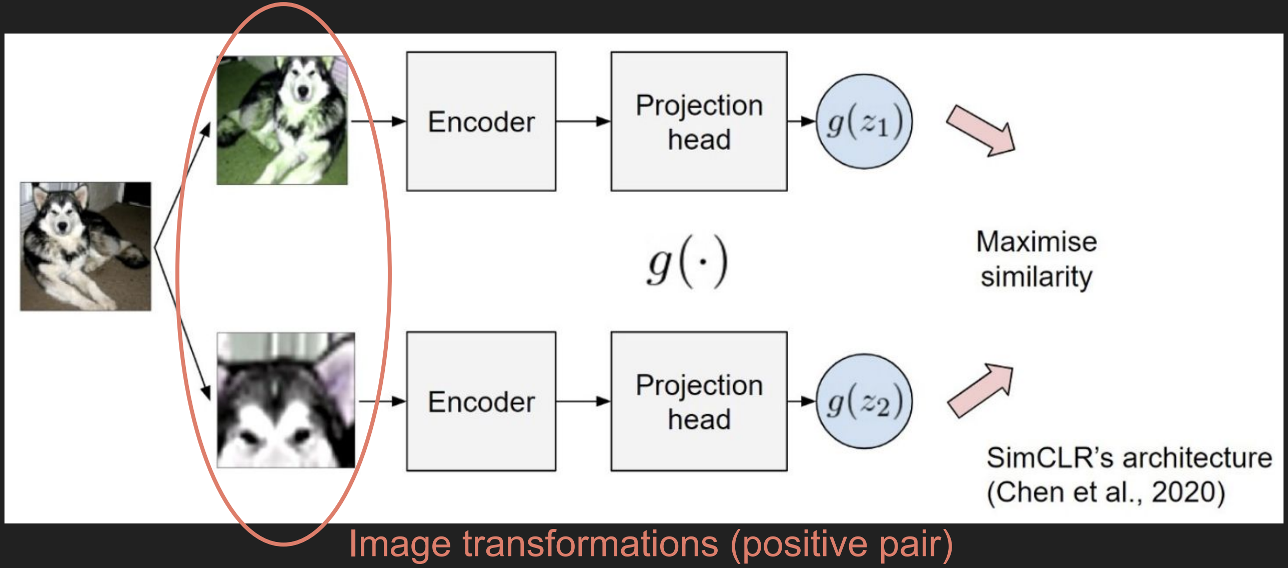


Positive pair

Negative pairs

Contrastive Representation Learning (from Vision to Code)

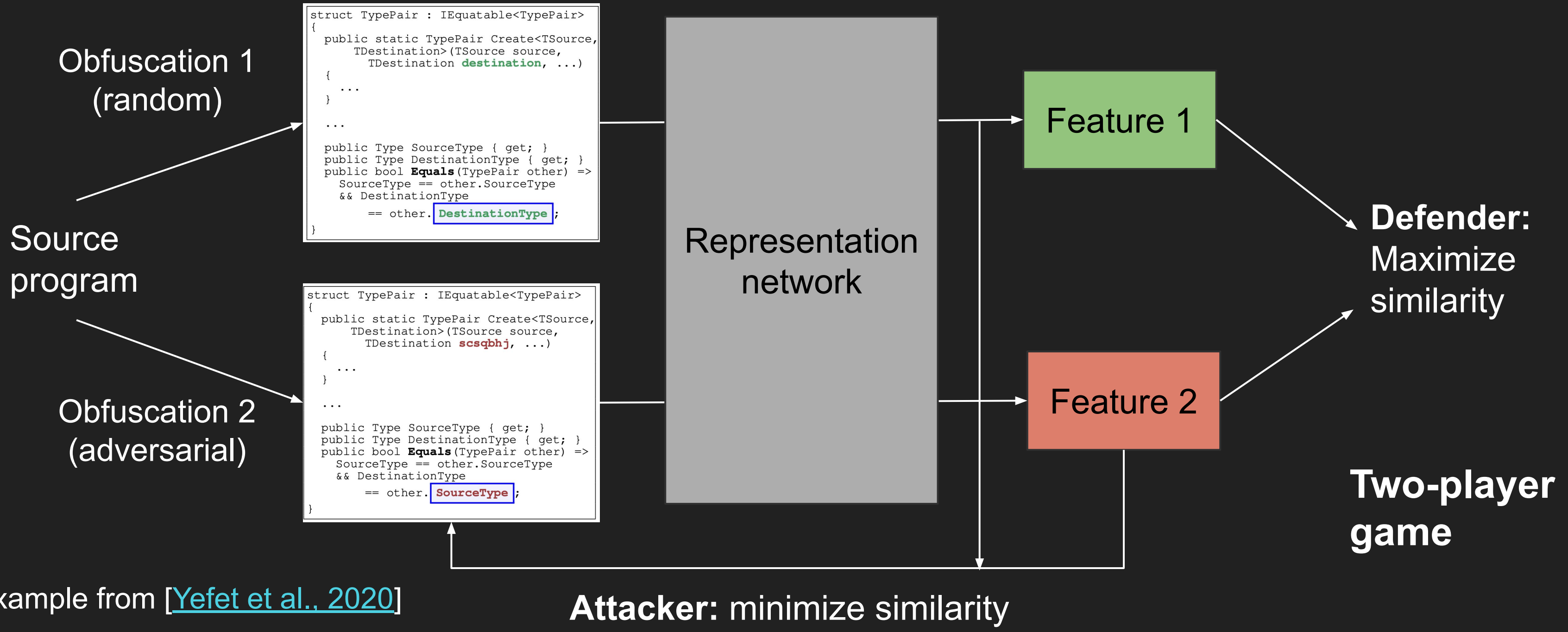
Contrastive learning: Learn representations by prompting data transformation invariance ([Chen et al., 2020](); [Foster et al., 2021]())



(SimCLR architecture; image from [TDS blog]())

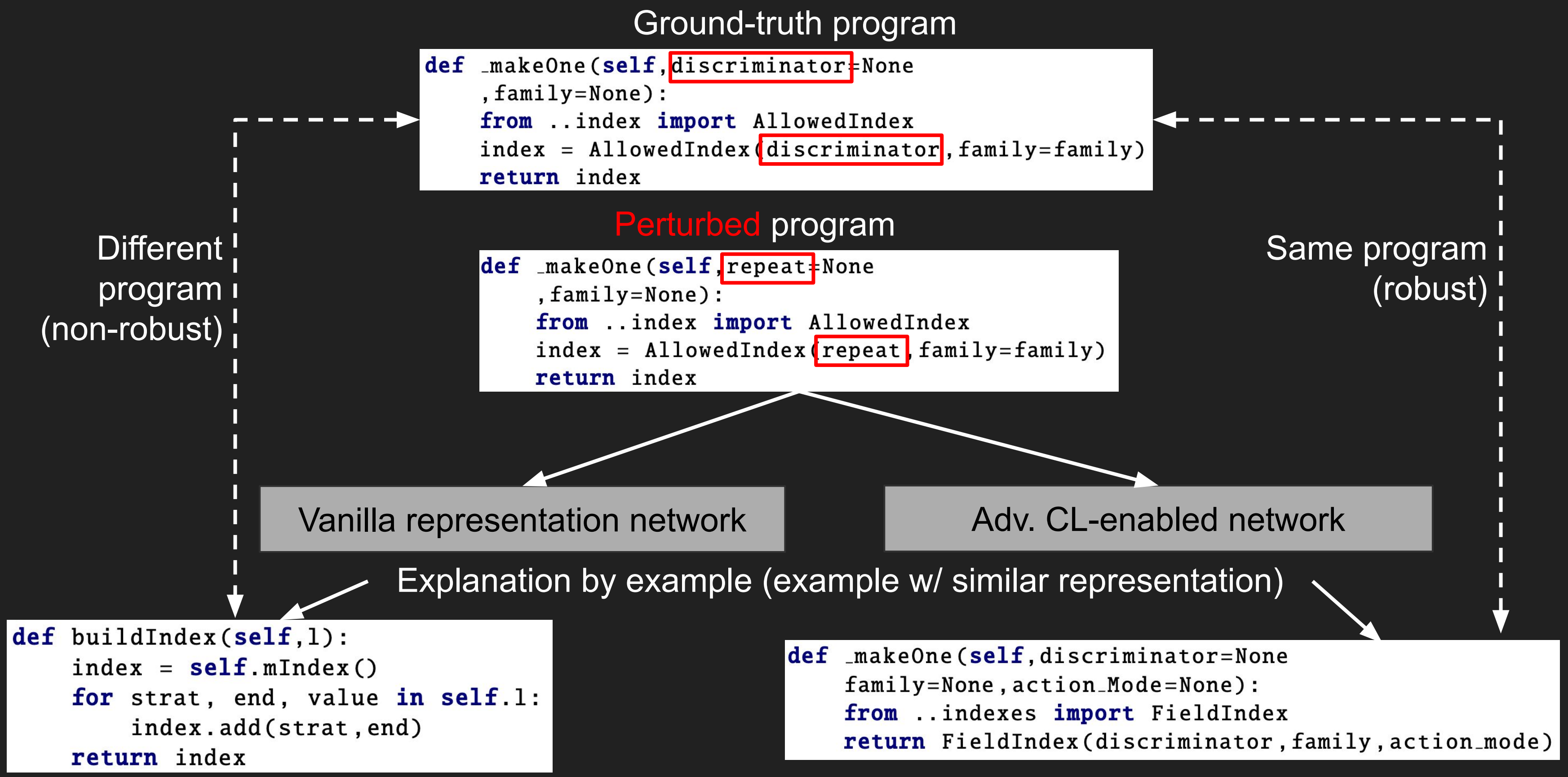
Robustness from ‘Adversarial’ Views of Code

- Regards adversarial code (worst-case obfuscation) and benign code as a positive pair, then contrastive learning enforces ‘robustness’ due to transformation ‘invariance’



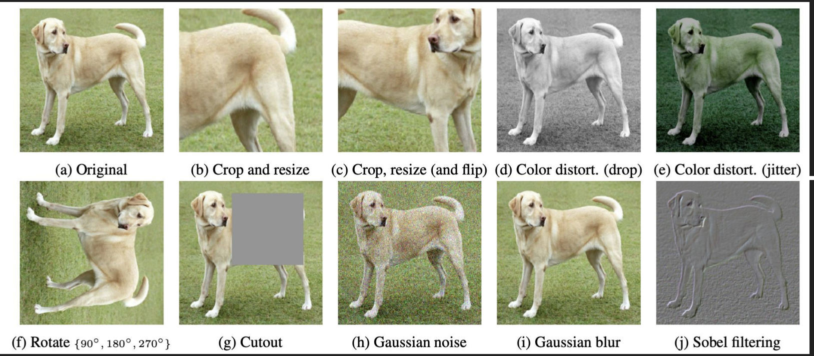
Code example from [Yefet et al., 2020]

Robustness Gain by Adversarial Code Contrastive Learning



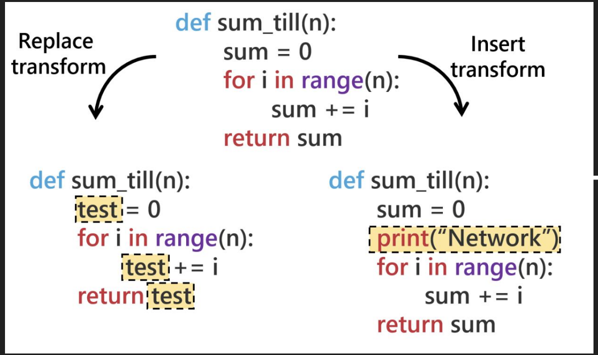
Contrastive Representation Learning (Vision & Code)

Image transformations (Chen et al., 2020)



Contrastive learning

Adversarial code transformations



Contrastive learning

Part 4

Hands-on Demo & Code Walkthroughs

bit.ly/nips-22-content

Overview

- Evaluation setup for vision models (image classification)
- Evaluation setup for code models (code summarization)

Modality I: Computer Vision

- Empirical evaluations of similar capacity models on the robustness benchmark datasets (image classification: top-1 accuracy, AUPR, mFR, mT5D).
- The datasets will cover different aspects like corruptions, perturbations, background dependence.
- Possible attribution factors of improved robustness.
    - Masking
    - Sensitivity analysis
    - Frequency spectrum
    - Loss landscape
    - Mean attention distance (relative receptive field)

Modality II: Code

- Empirical evaluations of similar capacity models on the task of code summarization.
- Record F1-scores of the models on clean examples and adversarial examples.
  - Random site-selection + optimal site-perturbation. [Ramakrishnan et al., 2020]
  - Optimal site-selection + optimal site-perturbation. [Srikant et al., 2021]

Thanks to Jinghan Jia (Michigan State University) for helping with this part.

Part 5

Concluding Remarks and Q&A

Take-Aways

- Foundational robustness: evaluation and enhancement of model correctness against natural and adversarial data shifts - a foundation of trustworthy AI
- The prevalence of foundation models also shift the focus of robustness from *task-centric* to *representation-centric*
- Lunch is still not free: Higher standard accuracy of downstream tasks using foundation models ≠ improved robustness
- Methods to evaluate and improve foundational robustness in pre-training and fine-tuning stages
- Robustness comes best with practice

Resources

- J. Z. Kolter and A. Madry: Adversarial Robustness - Theory and Practice (NeurIPS 2018 Tutorial)
- Pin-Yu Chen: Adversarial Robustness of Deep Learning Models (ECCV 2020 Tutorial)
- Pin-Yu Chen and Sijia Liu: Zeroth Order Optimization: Theory and Applications to Deep Learning (CVPR 2020 Tutorial)
- Pin-Yu Chen and Sayak Paul: Practical Adversarial Robustness in Deep Learning: Problems and Solutions (CVPR 2021 Tutorial)
- Pin-Yu Chen: Holistic Adversarial Robustness for Deep Learning (MLSS 2021 Tutorial)
- Pin-Yu Chen: Adversarial Machine Learning for Good (AAAI 2022 Tutorial)